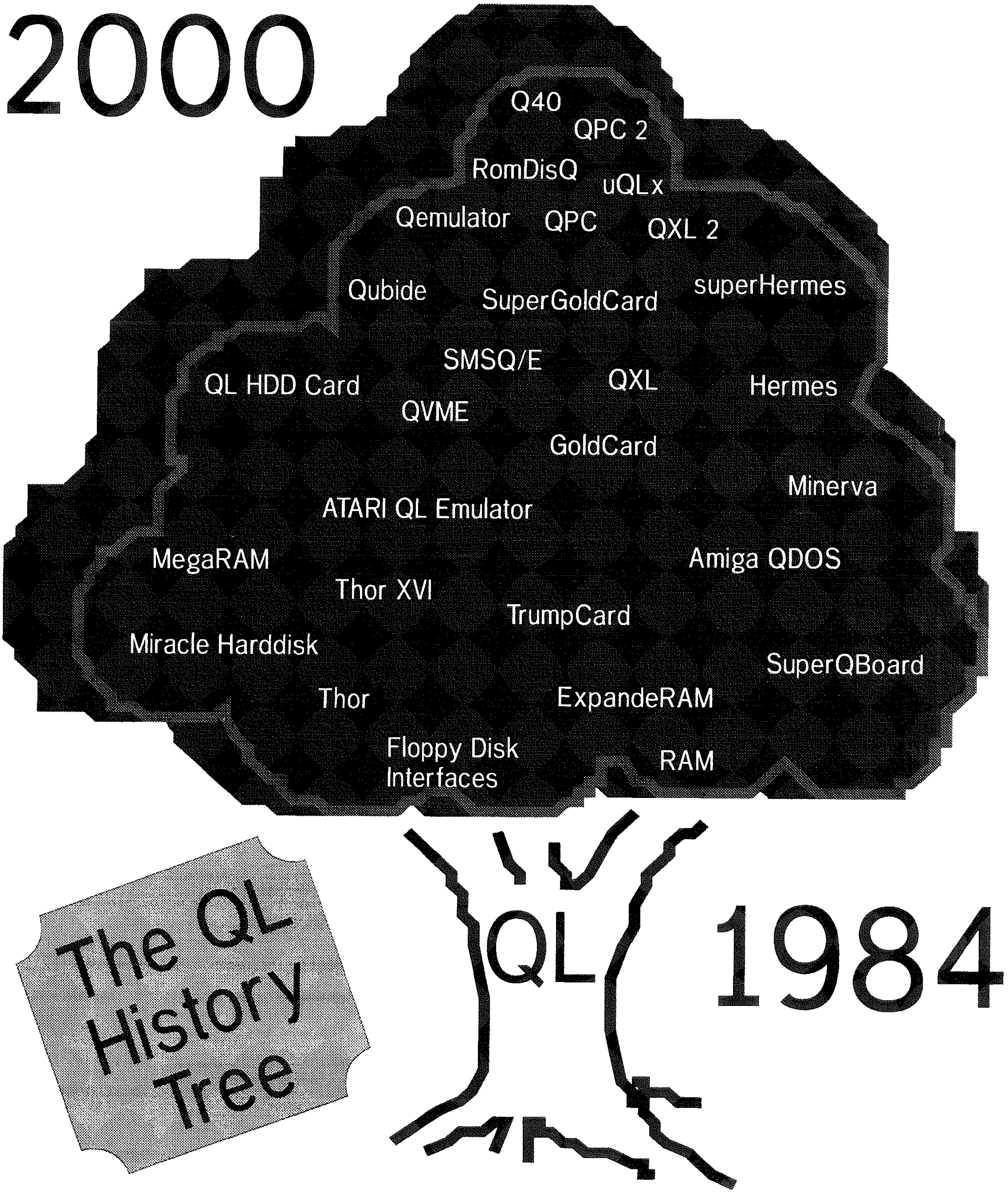
QL Today

Volume 4
Issue 5
Jan./February
2000

ISSN 1432-5454

The Magazine about QL, QDOS, Sinclair Computers, SMSQ...



Contents

Advertisers

in alphabetical order

QL *Today*

ISSN 1432-5454

German office & Publisher:

Jochen Merz Software	Tel. +49 203 502011
Im stillen Winkel 12	Fax +49 203 502012
47169 Duisburg	Box1 +49 203 502013
Germany	Box2 +49 203 502014
	email: JMerz@j-m-s.com

English office:

Q Branch	Tel. +44 1903 200212
The Bank Volt	Mobile +44 836 745501
6 Coronation Buildings	Fax +44 1903 208070
Ham Road	Fax +44 1273 381577
Sussex BN11 2NN	email: qbranch@qbranch.demon.co.uk
United Kingdom	

Editor:

Dilwyn Jones	Tel. +44 1248 354023
41 Bro Emrys	Fax +44 1248 354023
Tal-Y-Bont, Bangor	email: dilwyn.jones@dj.softnet.co.uk
Gwynedd	
United Kingdom LL57 3YT	

QL *Today* is published bi-monthly, our volume begins on 15th of May. Subscriptions begin with the current issue at the time of sign up. Please contact the German or English office for current subscription rates.
Payment in DM (drawn on a German bank) can be made by either Cheque or Euro-Cheque, Payment in £ (drawn on an English bank) can be made by Cheque. Cheques should be made payable to Jochen Merz Software (German office) or QBranch (English office).

Credit Card holders may subscribe by either calling or sending their Credit Card number and Expiry date. Credit cards will be charged in DM (German office) or in £ (English office).

We welcome your comments, suggestions and articles. YOU make QL *Today* possible. We are constantly changing and adjusting to meet your needs and requirements. Articles for publication should be on a 3.5" disk (DD or HD) or sent via Email or into one of the JMS-BBS's. We prefer ASCII, Quill or text87 format. Pictures may be in _SCR format, we can also handle GIF or TIF To enhance your article you may wish to include Saved Screen dumps. PLEASE send a hardcopy of all screens to be included. Don't forget to specify where in the text you would like the screen placed.

Article and Advertising deadlines are as follows:

Issue 1: 30 April	Issue 2: 30 June
Issue 3: 30 August	Issue 4: 30 October
Issue 5: 30 December	Issue 6: 29 February

Dilwyn Jones

Well, the QL made it safely into the year 2000, with only a few minor glitches in a small number of programs such as Payroll and QL Integrated Accounts, which affected Quanta - see Byts of Wood. The PBOX BBS system for the QL was affected although the author moved quickly to issue a patch.
I suppose it's traditional at the end of a decade to look back. More so at the end of a millennium, as we enter the 'noughties' as some people refer to the decade from 2000. So I decided to chronicle the last 16 years of QLing, listing the significant events I could remember or dig up. I found there's no shortage of news to report, as our news pages still testify! Did I omit anything important? Write and let me know!

Articles. This is an ongoing request - there must be someone out there prepared to write for the magazine. While am I very grateful to the regular contributors of course, it would also be nice to be able to include some new names - just send me an article on disk or by email or give me a ring to discuss possible subjects.
Don't forget the competition to create the best screen saver module for the CueDark program on the last issue's cover disk - as we have had very few entries so far, you'd stand a high chance of winning some free software from JMS!

In the year 2,000 we have a wider choice than ever of platforms on which to do our QLing. Whether we prefer an expanded original black QL, an Aurora or Q40 (the machine of the moment), or an emulator on another computer, there is no reason at all to stop using a QL. The colour drivers (or GD2 - see Dave Westbury's article) are here - perhaps in the tradition of contrived QL names we could call them QoLour Drivers?
Byts of Wood reports that Nasta is still keen to see his Goldfire expansion released if possible, despite the cessation of trading at Qubbesoft P/D back in December. TCP/IP is almost here - I'm waging a little campaign to try to get Quanta to consider financing its further development and possibly integration into SMSQ/E and QPC as I don't think we can be taken seriously in the 21st century without proper email and Web access from our systems, especially now that even the Spectrum and Z88 are leaving us behind in this respect! Support for my campaign will be most welcome - petition Quanta to support my campaign.

Where do you think that we go from here as QLers? Write in and let me know so that we can express opinions on the future of the QL in print so that those who hold the purse strings know where to direct the money.

MILLENNIUM DOME CONTROL CENTRE
Brilliant idea to install Q40s! Arise sir Bloggs!
Y2K PROOF
H.M.
TWO Q40s!!
YIPPEE A Q40!
DOME
LOOK! A QL!
DOME A-OK
BLOGGS Q40

NEWS

New Year - new QL Today layout? No, we're quite happy with our layout and the fonts and it seems, our readers feel the same way. News will be two-column to take account of the fact, that more and more internet URLs appear in the news. To be different, we left-justify the news this time.

Jean-Yves Rouffiac QL Web Pages

Jean-Yves Rouffiac, author of the QL adventure game Dreamlands, has sent us an email telling us about his Web site, which includes a copy of the Dreamlands game in QLAY emulator format. He says he is not sure if this format would work with the other QL emulators (can anyone tell us?). The URL to get to his site is:
www.westhaven.u-net.com

In time the site will be expanded to include other QL software and some OS/2 software he has written. He can be contacted by email at
jeanyves1999@netscapeonline.co.uk

NESQLUG Group Web Site Changes

The NESQLUG web site URL has changed. It is now http://boehm.home.hiwaay.net Note the two a's in hiwaay. The public section contains the freeware MIDIPlayer and other MIDI programs for the QL. Old reviews are also kept in the public section.

PBOX now Y2K Compliant

A new release of the PBox Bulletin Board System for the QL is available from Phil Borman's Web site to address certain Year 2000 compliancy issues. Version 1.22u can be downloaded from:
www.pborman.freeserve.co.uk/pbox122u.zip

GD2 Documentation

QBranch now have available an 11 page booklet from Tony Tebby with information on the Graphics Device Interface Version 2 (or GD2 for short) for programmers. The GD2 screen driver is a hybrid of the QDOS/SMSQ Extended Environment screen driver (as used on Q40 for example) which maintains a high level of compatibility with existing QDOS/SMSQ software and very few modifications to the internal data structures.
Subjects discussed include the use of 'wallpaper' pictures, palette maps, colour definition selection and notes on blobs and patterns under the new system.
It documents the use of the old QL style colours 0-7, palette mapped colours 0-255 (on hardware that does not have a true palette map, changes do not affect information already on the screen) and the 24 bit true colour modes.
The document lists both the SBASIC extensions and the system calls for assembler programmers. There is also a table listing the standard palette map predefined for colours 0-63, along with RGB colour values, VGA D.A.C. values, and 8/16/24 bit colour values.

Archivers Control Panel v4.00

A new ACP version (v4.00) is available from Thierry Godefroy's Web site at:
qdos.cjb.net/english/download.html
The changes since the last version are:
- tar/gzip/bzip2/compress support added.
- New "tools" sub-menu allowing to split large files into several fragments.
- Extended "Configuration" sub-menu.
- Partly re-written software with many small improvements (too many to be listed here) and some minor bug fixes.

New releases of tar, gzip and bzip2 are also available and 100% compatible with ACP v4.00.

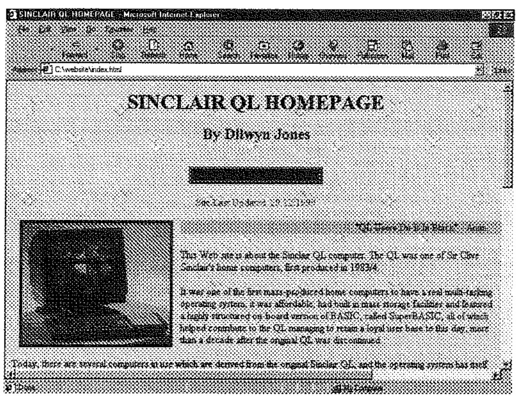
Chas Dillon Sources

Chas Dillon, the author of many QL programs in the 1980s, has now released the sources to many of his programs published by Digital Precision Ltd and PDQL in previous years.
The source files to Cash Trader, Payroll, Hardback, Media Manager, Turbo, BetterBasic, CrossRef, Compare, Editor, and EditPrint are now available from his Web site on
www.realcom.co.uk/ql_thor/index.htm
Some conditions are placed upon those who wish to download and use the sources - a link on the page above takes you to another page which lists conditions.
QL Today readers will know hopefully that a small team headed by Mark Knight is working on updating some of these programs, most notably Turbo and Editor. RWAP Software have enhanced versions of Cash Trader and Payroll - see their news item in this issue.

Dilwyn Jones' Website

Your kindly editor has just set up his own QL-related Web site. It contains information about QL

Today (of course), his software (of course) and even some pages about sources of products for the QL, a list of books published about the QL and even a potted history of the QL right up to the present day compatible systems, plus links to all sorts of other QL-related Web sites. You'll even find some QL-related humour pages in there!
If you have Web access why not drop in on:
www.soft.net.uk/dj/index.html



SINCLAIR QL HOMEPAGE

By Dilwyn Jones

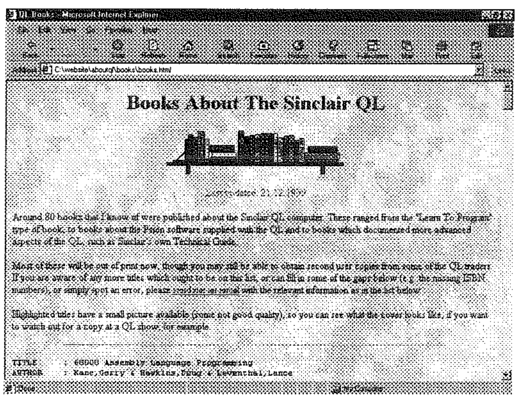
Site Last Updated 18.12.1999

"QL There Do It In Black" - Anon

This Web site is about the Sinclair QL computer. The QL was one of Sir Clive Sinclair's home computers, first produced in 1983/4.

It was one of the first mass-produced home computers to have a real multi-tasking operating system, it was affordable, had built in mass storage facilities and featured a highly structured on-board version of BASIC, called SuperBASIC, all of which helped contribute to the QL managing to retain a loyal user base to this day, more than a decade after the original QL was discontinued.

Today, there are several computers in use which are derived from the original Sinclair QL, and the operating system has itself


QL Books - Microsoft Internet Explorer
C:\website\aboutql\books\books.html

Books About The Sinclair QL

Last updated: 21.12.1999

Around 80 books that I know of were published about the Sinclair QL computer. These ranged from the 'Learn To Program' type of book, to books about the Psion software supplied with the QL and to books which documented more advanced aspects of the QL, such as Sinclair's own Technical Guide.

Most of these will be out of print now, though you may still be able to obtain second user copies from some of the QL traders. If you are aware of any more titles which ought to be on this list, or can fill in some of the gaps below (e.g. the missing ISBN numbers), or simply spot an error, please send me an email with the relevant information as in the list below.

Highlighted titles have a small picture available (some not good quality), so you can see what the cover looks like, if you want to watch out for a copy at a QL show, for example.

```
TITLE      : 68000 Assembly Language Programming
AUTHOR     : Kane,Gerry & Hawkins,Doug & Leventhal,Lance
```

Micro EMACS 4.00 Update

An updated MicroEMACS v4.00 release (03/01/00 release) is available from Thierry Godefroy at the following URL:
qdos.cjb.net/english/download.html
The changes since the last release are:

- A bug corrected dealing with buffer allocation and triggered when syntax highlighting was enabled by default (e.g. in uelocal_rc) and a filename was passed in the command line of MicroEMACS.
- MicroEMACS is now linked with libqmenu_a v1.05 and is therefore Q40-compatible.

As a result the "EMACS_Q40BUG" environment variable is no more taken into account. Many thanks must go to Tim Swenson who did the initial bug report as well as a lot of beta-testing on the Q40 for me.

QL-2-PC Transfer Demo

JUST WORDS! has revised its demo disk to include a demonstration version of QL-2-PC Transfer. This is a fully working version of the program, but has a restriction that only the first 3,000 bytes of a file (about 300 words) can be transferred.
The demo disk now contains only the pointer versions of the JUST WORDS! program range.

Z88 User Magazine

Issue 5 of Z88 User magazine should be available by the time you read this. It carries details of the TCP/IP stack system for the Z88, which editor Darren Branagh now claims leaves the QL as the only major Sinclair machine without Web access facilities of this kind - he knows of several people in Britain and the USA accessing the Web and sending him emails via a Z88. Issue 5 contains 36 pages, the biggest issue yet, full of reviews. Z88 User is available from W.N.Richardson & Co.

Q Branch News

We have just released **The Knight Safe 3**. This new version has a number of new features and now splits the 'Save' and 'Restore' functions of the program into two separate modules making both sections faster to use. The 'Restore' section also now allows you to retrieve single files from an Archive. The price remains the same and upgrades from previous versions cost 5.00 pounds. (New Manual supplied)
I am currently testing the beta versions of the colour drivers for the Q 40. These seem to be very stable and able to run most of the programs I have tried.
Two current problems are with Text 87 and The Lonely Joker but we are looking into the reasons behind this. Text 87 may need a new mode specifically for its way of using the display since it is unlikely that the author will be upgrading the program. Any Q 40 owners who would like to take part in the beta testing should contact Q Branch or send a stamped addressed A 4 sized envelope for the disk and the manual.

PROGS have released new versions of some of the ProWesS drivers to work with the new colour drivers on the Q 40. Anyone wishing for upgrades to ProWesS should return the master disk with a S.A.E.

Q Branch will be handling the new 'Agenda' program from Jochen Merz (22.00 pounds) and will be able to provide v 1.53 of QPC 2 to customers wishing for an upgrade. (send master disk and S.A.E.)

News from Mark Knight

Bringing Turbo and Editor into the new century. Since Chas Dillon has now released the source code to Editor and Turbo and a lot of other QL software I have been very busy. First I have been checking with some beta test volunteers that they are still interested in a new version of Turbo: next I checked that the programmers were still interested in updating it: then I begun planning how the Turbo update will be done. The Turbo update work will begin in earnest in the new year, programmers and beta test team permitting.
As the task is within my programming abilities I decided to begin updating the Editor myself. Version 2.05 of Editor has been around for many years and after patching it can even be persuaded to take advantage of high-resolution screens. There are problems with long filenames though and using a separate patch program to force Editor to use large screens is far from ideal.
Work on a new version of Editor is progressing rapidly and unless something unexpected happens I should release it around the middle or end of February 2000. When I obtained the source there were two versions, 2.30 and 2.10, both in SuperBASIC and compiled form. 2.30 contained some serious bugs though and I decided to roll back to 2.10 and start my update there. This ended several days of frustration and fruitless bug-hunting so it turned out to be a good decision. All the features found in 2.30 were present in 2.10 anyway, the only significaant advantage of 2.30 was faster sorting, and a good look at the sort persuaded me I could speed it up further in any case.
At the start of the 2.10 source code were a large number of REMark lines listing known bugs and I am happy to report that I have fixed all those in that list. I later added six bugs to the list from my own knowledge, of which I have already been able to fix three at the time of writing.
When producing 2.10 Chas Dillon added several useful new features including: a qualifier to prompt in the command line for filenames: a command to put the current line of the file into the command line: a flag in the status line to indicate that a file has changed since loading/saving: and a lovely new command to add up a colum of figures and insert the total into the text at the cursor position. He also speeded up file loading in some modes which is most welcome, though it wasn't slow before.
I have so far added the following: improved column addition facility; faster sorting; three different sort modes with new commands to switch between them; descending sort capability as well as the original ascending; updated help file and manual (manual is an Editor document); several new cursor movement commands; proper support for high resolution screens; long filename support in Editor and the configuration program; and the ability to configure Editor in a subdirectory. One version I produced recently has been tested on Aurora, QPC-2 and the dreaded Atari TT and it worked equally well on all these platforms as well as on my humble Gold Card QL.
I plan to try and fix more bugs and add more speed to the sort, and to perhaps improve the support for high resolution screens still further. The release date is so far away (it's December as I write) because I'm determined to do a good job of updating Editor, and I won't rush to distribute untested code. I will bundle the source code with the program as I supply it, and I hope that others will also fix bugs and let me know about the fixes.
The new version of Editor will be supplied to Phil Jordan's QL public domain library as FreeWare and will also be available from other FreeWare sources.
Mark Knight, 304, Portobello Road, Notting Hill, LONDON, W10 5TA.
Telephone (020) 8932 6987 (Not after 9:30 p.m. or before 9:00 a.m.)

CD-WRITING WITH Q40

Q40 users who are into Linux can now use a CD-Writer with their machine thanks to some nifty work by Richard Zidlicky. A report on Claus Graf's Q40 Web site **www.Q40.de** says that a Philips CDD-3610 writer has been successfully tested on a Q40 running Q40-Linux, using SCSI software modules for Linux downloadable from Richard Zidlicky's ftp site
ftp://ftp.uni-erlangen.de/pub/linux/680x0/q40/
Single and double speed writing has been achieved using the above drive via a SCSI interface.

News From Jonathan Hudson

Sox Sound Utilities for QDOS
Sox has been ported to the QL by Jonathan Hudson and made available from his Web site as of 03/01/2000. This is a package of sound conversion utilities for QDOS, which includes source, documentation and binaries. Q40 users may find it useful for converting other sound file formats to the _ub (unsigned byte) sound file format of the Q40, which just contain uncompressed 8 bit sampling data. 1st byte right channel, 1st byte left channel, 2nd byte right channel, 2nd byte left channel and so on. Version 12.16q0 implements sox 12.16, gsm 1.0-pl10, and a later vcnvt plus some better examples.

WXQT2
wxqt2 0.02 (600Kb, 06/11/99) is a graphical front end for qltools 2.14 and qxltool 1.10. v0.02 is the first public release.
This program uses the wxwindows C++ class libraries to build binaries for both Unix/X and Microsoft Windows from the same source code. Included all source, documentation and binaries (WIN32).
Selected files can be copied using the left/right arrow. A popup menu offers options to view, delete, mkdir, format, disk info etc. Support for floppy disks, native file format (e.g. DOS or Windows files) and QXL.WIN files.
QFM
qfm 0.08 (c.55Kb, 03/01/00) is a menu driven front end for qfax and lfax. Binary, source and documentation. Requires menu_rext and hot_rext. 0.05 Includes workarounds for SMSQ/E mis-features, fixes the 'seletion key' bug and adds a "faxprint" example for a more friendly fax print function. 0.08 linked with contemporaneous qmenu. Works on Q40.

Jonathan Hudson's Web site is on:
http://home.freeuk.net/diagon.alley/index.html

QL HACKERS JOURNAL

Issue 32 of QL Hackers Journal should now be available from Tim Swenson's Web site
www.geocities.com/SiliconValley/Pines/5865

RWAP Software News

A demo version of FlightDeck is now available from me, cost £2. This limits flying time to 10 minutes.
SToQL v1.30 has now been released. This version works on all QL high resolution screens. QL Payrollv3.5 and QL Cash Trader v3.4 have now been released which enable QL owners to produce trading accounts and payrolls (and all UK tax/national insurance reports) required by a small busines. Cost is £5 each (either as an upgrade to earlier versions or as new copies).

CueDark Competition

Here is the first (and only, so far) entry from Dietrich Buder:

Moving digital clock

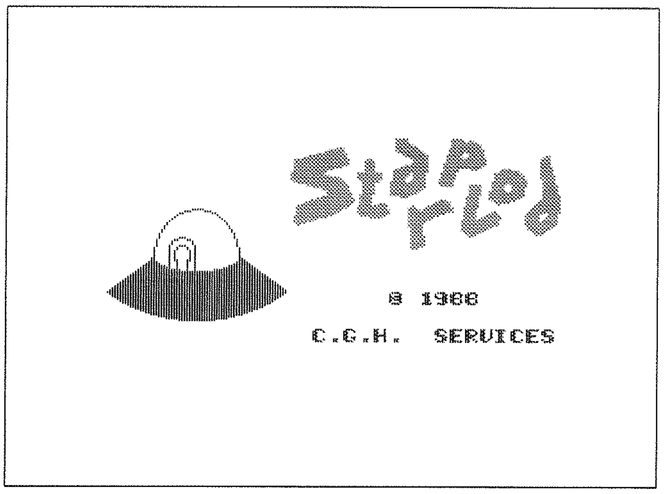
```
1000 REMark ** DKL_DATUM_BAS ** - 09.12.1999 -  Dietrich Buder - 07.12.1999
1010 :
1020 REMark Moving digital clock for 'CueDark'
1030 :
1040 REMark QLIB_V3.36 Input: LIBERATE 'dev1_DKL_DATUM','-NOLINE'
1050 DEF_INTEGER rx,ry,p,x,xlim,y,ylim
1060 :
1070 IF VER$='HBA': x=SCR_XLIM: y=SCR_YLIM: ELSE x=512: y=256
1080 WINDOW x-2,y,0,0: CLS
1090 OPEN #3;'scr': WINDOW #3; x-4,y,0,0: PAPER #3;0: INK #3;7: CLS #3
1100 t$=DATE$
1110 xlim=x-14-(6*LEN(t$)): REMark max. x-coordinate with CSIZE 0,0
1120 ylim=y-10:             REMark max. y-coordinate with CSIZE 0,0
1130 :
1140 x=0: y=RND(0 TO ylim): REMark start left at random y-coordinate
1150 rx=2:                  REMark arc
1160 IF RND(0 TO 1)=0: ry=1: ELSE ry=-1: REMark random direction
1170 :
1180 REPeat prog
1190  t$=DATE$: CURSOR #3;x,y: PRINT #3;' '&t$&' ': PAUSE 50
1200  PAUSE 50:            REMark 'PAUSE 1' does not work correctly
1210  x=x+rx: y=y+ry:      REMark new coordinate
1220  IF x<0: x=0: rx=-rx: REMark logic for new x and y
1230  IF y<0: y=0: ry=-ry
1240  IF x>xlim: x=xlim: rx=-rx
1250  IF y>ylim: y=ylim: ry=-ry
1260 END REPeat prog
```

Looking forward to other entries!! Don't forget that the competition is not yet closed!
There's still time to enter to win a free program from JMS!

Adventures on the QL - Part 5 : Starplod

Darren D. Branagh

First of all, let me start by apologising for the lack of an adventure review in the last issue. This was due partly to the fact that I was extremely busy at work and the fact that Dilwyn was short of space anyway (at least that's my excuse and I'm sticking to it ;-))
Anyway, as promised, we're taking a look at something totally different this time, as it's the final part of our look at QL adventures, its time for something special - what can only be described as an icon-driven pointer adventure!! It's the excellent STARPLOD, Again by Alan Pemberton, author of The Voyage of the Beano and of SQLUG fame. I have never seen anything like it on the QL before or since, and it remains unique to this day.



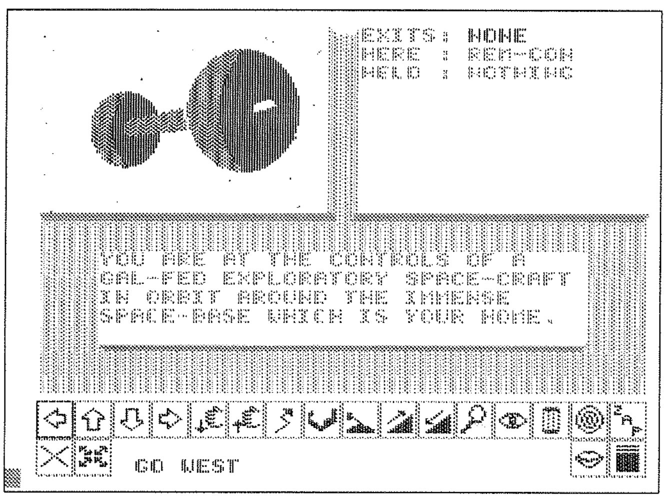
STARPLOD is found on the second of the ADVENTURE 93 disks, available from the PD suppliers and the QUANTA library (Voyage of the Beano is on the first disk, and another 3 other adventures are on the second disk too) So it is excellent value for money. On booting the 2nd disk, it gives you the option of selecting any of the 4 games on it by keying the first letter of the title - Therefore, we press 'S' for STARPLOD, and off we go...

Scenario

The scenario is this - you are a space traveller on a gal-fed exploratory spacecraft, in orbit around an immense space station you call home. It's your job to help the neighbouring planet colonies, with their requirements. If the planet Kevlar needs some uranium, it's your job to get it, all the time watching your backside against the bad guys, who are intent on taking over.

Layout

The screen is very colourful, and graphics are quite good. The screen is broken up into 4 main areas, the top left corner show a graphic of your current position in space - usually a picture of the planet you are circling above or any space ship which is near. Top right corner show statistics - such as any objects that are near or you're carrying, or any exits, and is also used to select a place to HyperDrive to (more on this later). The middle of the screen displays the current description of your surroundings, and all the available icons are at the bottom of the screen.



EXITS : NONE
HERE : REM-CON
HELD : NOTHING

YOU ARE AT THE CONTROLS OF A
GAL-FED EXPLORATORY SPACE-CRAFT
IN ORBIT AROUND THE IMMENSE
SPACE-BASE WHICH IS YOUR HOME.

GO WEST

Gameplay

Gameplay takes some getting used to initially, as it is not a text entry control system, as with other adventures. You move around and do things by means of 20 (yes, twenty!) different icons at the bottom of the screen. The four arrow icons move you north, south, east, or west for example. The ones of a flight of stairs with up and down arrows on them are used to 'ASCEND' and 'DESCEND' certain things - you get the picture (no pun intended!)
You can ANALYSE and EXAMINE just about everything in this game - the atmosphere around the planets you are trying to land on, the composition of the planets themselves, the spaceships, consoles, and any item up there you come across. The method for examining an object is very clever - bet you where wondering how to enter an object name to examine when there is no text entry? Well, all you need to do is click the ANALYSE or EXAMINE icon, analyse is used mainly for planets, examine being for objects. Then use the arrow keys on the keyboard to scroll through the text description in the middle of the screen until you highlight the text of the

JOCHEN MERZ SOFTWARE

Im stillen Winkel 12 D-47169 Duisburg
Tel. 0203 502011 Fax 0203 502012
http://www.j-m-s.com/smsq/index.htm

Jochen Merz Software Applications and Version Numbers

QPC1 + QPC2 + SMSQ/E Software QL-Emulator for PC's	[V1.52]	**DM 249,-**
QPC1 + QPC2 + SMSQ/E for SMSQ/E owners	[V1.52]	**DM 199,-**
QPC2 Upgrade from QPC1	[V1.52]	**DM 79,90**
SMSQ/E for ATARI or QXL or (Super)GoldCard	[V2.93]	**DM 249,-**
QD 98 Pointer-Environment-Editor	[VA.06]	**DM 119,-**
QD 98 Upgrade from V9	[VA.06]	**DM 39,90**
QD 98 Upgrade from previous versions	[VA.06]	**DM 49,90**
NEW! AGENDA for WMAN or ProWesS	[V1.02]	**DM 59,90**
QMAKE Pointer-driven MAKE for GST/Quanta Assembler	[V4.25]	**DM 45,-**
BASIC Linker	[V1.12]	**DM 49,-**
WINED Floppy/Harddisk Sector- & File-Editor	[V1.21]	**DM 49,-**
FiFi II File-Finder - Extremely useful!	[V4.19]	**DM 49,-**
FiFi II Upgrade from earlier versions	[V4.19]	**DM 19,-**
EPROM Manager	[V3.02]	**DM 49,-**
QSpread99 Spreadsheet Program	[V2.05]	**DM 139,-**
QSpread99 Upgrade	[V2.05]	**DM 39,90**
QPAC I Utility programs	[V1.07]	**DM 59,-**
QPAC II Files, Jobs & other Things	[V1.39]	**DM 119,-**
QPAC II Update	[V1.39]	**DM 16,-**
QTYP II Spell checker	[V2.17]	**DM 79,-**
QMENU The Menu Extension Programming Manual	[V7.04]	**DM 41,90**
QMENU Upgrade	[V7.04]	**DM 18,-**
QPTR Pointer Toolkit	[V0.30]	**DM 89,-**
QPTR Update	[V0.30]	**DM 16,-**
DISA Interactive Disassembler	[V3.04]	**DM 89,-**
DISA Upgrade from previous versions	[V3.04]	**DM 33,-**
typeset-ESC/P2 text87 driver for all ESC/P2 printers (incl. Stylus)		**DM 65,-**
CueShell	[V2.14]	**DM 90,-**
CueShell for QPC	[V2.14]	**DM 40,-**

Even if you already own QPC, you can upgrade to include CueShell for DM 40,-!!!

SER Mouse software mouse driver for serial mice		**DM 25,-**
EASYPTR Part 1&2	[V3.10]	**DM 89,-**
QDOS/SMSQ Reference Manual		**DM 79,-**

TERMS OF PAYMENT

Postage and package **[Germany]** DM 8,99 (if total value of goods is up to DM 50,- then only DM 5,99). **[Europe]** DM 14,50 (if total value of goods is up to DM 50,– then only DM 9,50). **[Overseas]** between DM 14,50 (1 item) and DM 35,– (maximum). All prices incl. 15% V.A.T. (can be deducted for orders from non-EEC-countries). E&OE. Cheques in DM, £'s, Eurocheques and Credit Cards accepted.

EUROCARD. VISA MasterCard

item to examine. I found this excellent and dead easy to use - no typing needed, you just use the arrow keys, and then the space bar to select the item! This is without doubt one of the cleverest and unique game control interfaces I have ever used, it's brilliant!

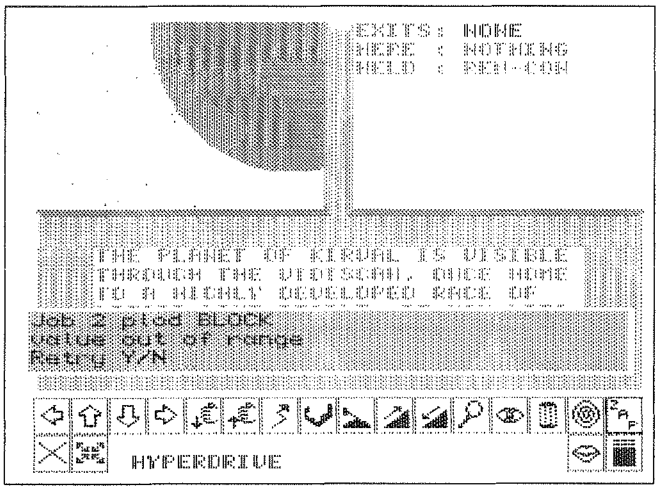
Moving around your ship or other ships (i.e. walking) is done with N, S, E, W arrow icons, but you move around the galaxy using the HYPERDRIVE and TRANSPORT icons; selecting hyperdrive brings up a map of the galaxy in the top right corner. Simply use the arrow keys again to move the highlighter to the planet on ship you want to hyperdrive to, and press Space - you're there instantly!

Transport allows you to beam down from your ship to the surface of a planet or onto another ship, and vice versa - Beam me up Scotty!

Another icon that is worth a mention is the GAL-PHONE icon. This allows you to communicate with all the other ships and planets in the area, and is used in much the same way as Hyperdrive, except you don't actually go there. Useful if you want to tell someone something important, but can't leave your present predicament, which is usually being shot at by the enemy - play it, you'll see what I mean.

Problems

Ah, here we go. There are a few problems running STARPLOD, namely that I couldn't get it to run properly on QPC1 or QPC2. When I tried to select icons at the extreme right of the screen Starplod would crash with an out of range error (see screenshot), although everything else would work perfectly. I have tried this on all resolutions from the standard 512x256 up to 800x600, and the results are the same. I can only conclude that it is incapable of running on QL's with higher screen resolutions. It runs just fine on a Standard black box QL with either a Trump Card or a Gold Card. I didn't try it on my QXL, and I don't own an Aurora but I expect the results would be the same as with QPC.

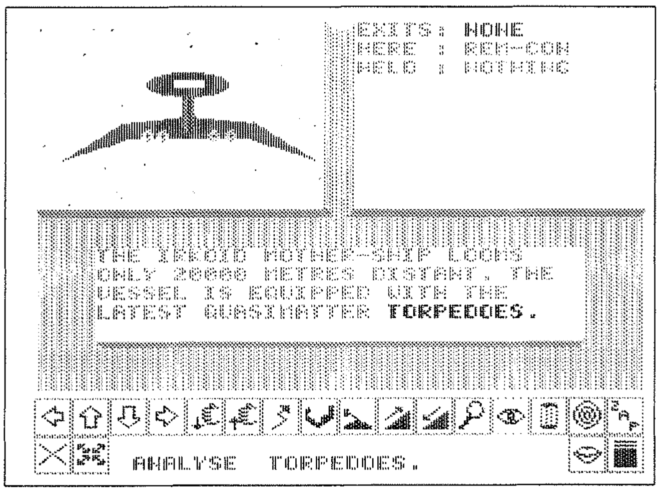

EXITS : NONE
HERE : NOTHING
HELD : REM-CON

THE PLANET OF KIRUAL IS UISIBLE
THROUGH THE UIDISCAN, ONCE HOME
TO A HIGHLY DEUELOPED RACE OF

Job 2 plod BLOCK
value out of range
Retry Y/N

HYPERDRIUE

The second grumble is the lack of instructions - There is a file called ADV93_DOC on the disk, but this just gives basic info on the entire collection of games and it is worth mentioning here that this Program is SQLUGware, written or the Scottish QL user group, and therefore registration


EXITS : NONE
HERE : REM-CON
HELD : NOTHING

THE IKKOID MOTHER-SHIP LOOMS
ONLY 20000 METRES DISTANT. THE
UESSEL IS EQUIPPED WITH THE
LATEST QUASIMATTER TORPEDOES.

ANALYSE TORPEDOES.

is advised I am unsure if this is still the case, so a note to the SQLUG lads might be a good idea if you intend to use STARPLOD. It was only from playing around with Starplod I managed to get the controls, and the scenario. I got my copy from Ron Dunnett's Qubbesoft Service, so it's unlikely I'm missing a _DOC file of instructions, but I suppose its possible.

It's also worth saying that you don't need the pointer environment to play Starplod (i.e. the files PTR_GEN or WMAN), as its icon system is built into the game. Therefore, sadly it doesn't work with a Mouse, and this is one area where I would love to see it updated - being able to use the mouse to point and click the icons would be so much better - Still, it was written in the late 1980's after all.

In Conclusion

I enjoyed Starplod, and wasn't put off by the gripes, although I use QPC2 as my main QL now, so lack of use on that is frustrating (anyone care to update it for use on QPC please?) and the lack of instructions was not a disaster, as I hate reading manuals anyway and nearly always wade in at the deep end first and refer to the manual only if need be, thankfully RTFM (Read The F***ing Manual) ***[Flaming? - Editor]*** didn't reign supreme here.

As I said earlier Starplod is unique in QL games, and is the nearest you'll get on the QL to the fab role playing 'point and click' adventures you'll find on the PC or Playstation. As such, it should be embraced as a triumph. I hope you enjoyed our little look at the Adventure world in the last handful of Issues of QL Today, and that it encourages you to try one or two on your own QL - go on, you might even like it!

QUANTA

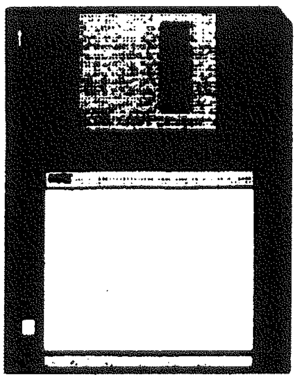
Independent QL Users Group

Worldwide Membership is by subscription
only, and offers the following benefits:
Monthly Newsletter - up to 40 pages
Massive Software Library - All Free !
Free Helpline and Workshops
Regional Sub-Groups. One near you?
Advice on Software and Hardware problems
Subscription just £14 for UK members
Overseas subscription £17

Barclaycard: Visa: Access: Mastercard

*** Now in our SIXTEENTH successful year ***

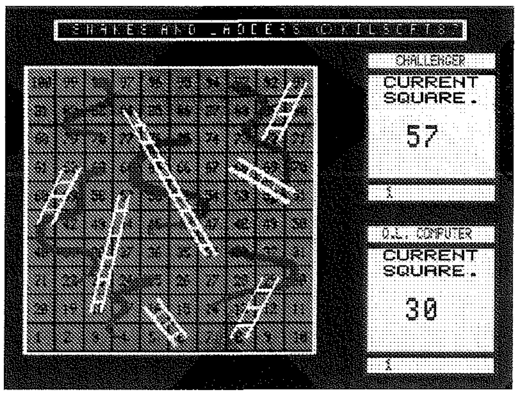
Further details from the Membership Secretary

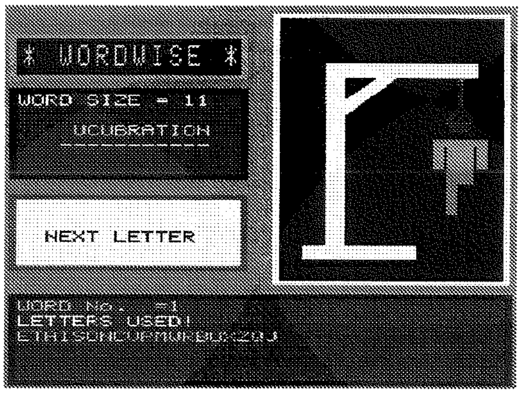


Bill Newell
213 Manor Road
Benfleet
Essex
SS7 4JD
Tel. (01268) 754407

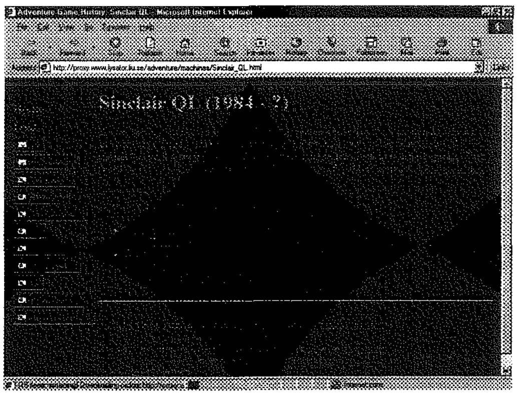
More QL related Web Sites

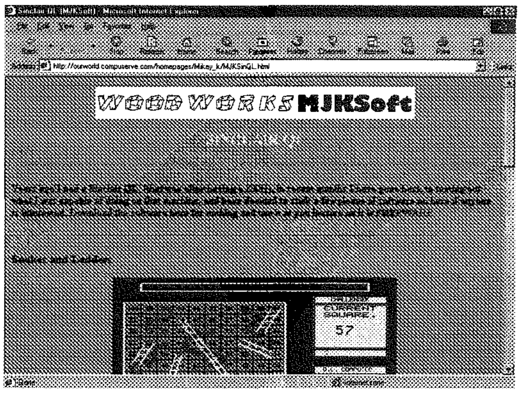
Mike Kilmister of MJKSoft maintains a Web site called "Wood Works" with some QL software available for download, including a Snakes and Ladders game, Hangman (Wordwise) and a simple Astrology program. "Wood Works" is on:
`http://ourworld.compuserve.com/homepages/MikeyJ/MJKSinQL.html`





Another Web site with QL-related content - mainly adventure gaming - is on
`http://proxy.www.lysator.liu.se/adventure/machines/Sinclair_QL.html`





Nick Cheesman is a long time QLer who returned to programming a QL after becoming a Visual BASIC programmer. His Web site includes some QL-related material and articles and can be found on:
`http://www.netcomuk.co.uk/~njc1/welcome.htm`
The site includes a NetZine called MonoCall which runs some QL listings and articles.

CAUTION: HOTKEYS - Don't Burn Your Fingers - Part 2

David Denham

David Denham shows how to tame some less than perfectly behaved programs with hotkeys.

I promised that in this instalment I would describe variations on the hotkey definitions introduced in the first issue to help tame the habits of software like Quill.

Before I move on to that, I'd like to introduce some helpful commands and functions for using hotkeys. These list the definitions you have set, and tell you what type of definition is attached to a particular key.
The most useful command is HOT_LIST. As its name implies, this gives you a list of keys already defined. See Fig. 1 for an example.

This command lists by default to channel #1 on the screen, but if you specify the output channel, it will list to there instead. In common with some

Figure 1

ENTER	KEY last line recall
SPACE	KEY
.	WAKE Button_Pick
/	KEY ed qfind
A	FLP1_Archive,Archive
C	EXEC Calculator
F	WAKE Files
K	EXEC Calendar
L	CMD Liberate
Q	LOAD FLP1_Quill,Quill
a	PICK Archive
b	PICK
c	PICK Calculator
f	PICK Files
k	PICK Calendar
l	PICK QLib_3.36
q	PICK Quill
x	WAKE Exec
cF1	WAKE button_sleep
sSPACE	KEY

Toolkit 2 commands, you can even make it list to a file if you really insist by using a backslash and filename in place of a hash symbol and channel number. This was how I created the list in figure 1. I am not sure if this will work in older versions of the hotkey system, as it does not seem to be documented in my copy of the manual.

`HOT_LIST`	lists to channel #1 by default
`HOT_LIST#2`	lists to channel #2
`HOT_LIST \ram1_text`	lists to file called ram1_text

The command only lists just enough information for you to remember what was on that key - it does not list the special variations on some types of definitions.

The name of a program associated with any particular hotkey definition can be returned by a function called HOT_NAME$. Using my definitions of Figure 1, I could check which program was associated with the Q key with the command
```
PRINT HOT_NAME$('Q')
```
which would return FLP1_Quill. Note that because different definitions can go on upper case and lower case keys, HOT_NAME$ can also return different values for either case. So PRINT HOT_NAME$('q') would return Quill in this case. If the key requested is not defined, you get an empty string "" back.
The HOT_TYPE function returns a number which tells you what type of definition has been assigned to the key given. If it returns a value of -7 (code for the Not Found error message), it means that the key you specified does not have a definition. Some of the definitions created by 'the system' have negative HOT_TYPE values:

-8	last line recall (ALT ENTER)
-6	stuff keyboard queue with previous stuffer string (SHIFT ALT SPACE)
-4	stuff keyboard queue with current stuffer string (ALT SPACE)
-2	stuff keyboard queue with given string
0	pick BASIC and stuff command (HOT_CMD like L in Figure 1)
2	do code
4 or 5	execute a Thing
6	execute a file (like Q in Figure 1)
8	pick a job program (like lower case q
in	Figure 1)
10 or 11	wake or execute a Thing (e.g. a QPAC2 menu)
12	wake or execute a file

I mentioned HOT_GO and HOT_STOP in the last issue. In fact, there are other commands to suspend individual hotkey definitions, reactivate that key, or even kill off a key's definition.
Suppose I wanted to prevent myself accidentally pressing ALT Q to start another copy of Quill by mistake. Once I have started my main copy of Quill, I can use the HOT_OFF function to 'suspend' ALT Q:
```
ERT HOT_OFF ('Q')
```
I could turn it on again with the HOT_SET extension:
```
ERT HOT_SET ('Q')
```
More drastically, I could reassign the definition to a different hotkey. I could move the command to pick Quill (lower case q in Figure 1) to the 'u' key instead:
```
ERT HOT_SET ('u','q')
```
In essence, this defines ALT u as what ALT q used to be. The definition of ALT q is removed in the process. Not a facility you'll use that often, but worth remembering all the same.
You can kill off a definition altogether with the HOT_REMV function. ERT HOT_REMV('Q') will remove the ALT Q definition altogether, so it is not possible to accidentally try to load Quill from a non-existent disk or whatever. In some cases, you will need to use HOT_REMV on a key before you can reuse it for another definition. Note that if a program has been loaded into the common heap with HOT_CHP then that program is also removed from the heap as well as the key definition being removed.

Note the HOT_PICK definition on key b in Figure 1 above. This has no name. It is worth knowing that the SuperBASIC interpreter can be called a null string as far as hotkeys are concerned. If you tell a hotkey to pick "" (pick nothing as far as program names are concerned) it will try to pick SuperBASIC. As I am not an SMSQ user, I do not know how BASIC on SMSQ treats this convention.

[SBASIC daughter jobs are called SBASIC, but they can be renamed with the JOB_NAME procedure, so that individual PICK's are possible - Editor]

One other thing worth remembering is that you can tell BASIC to perform the action associated with a key, using the HOT_DO command. It is difficult to think of a practical example for this, so to illustrate what I mean, HOT_DO q entered as a command from SuperBASIC will do exactly the same as pressing ALT q if you had defined ALT q in your boot program to pick Quill.

Misbehaving Programs

Some older QL programs like Quill have some rather naughty habits. Either they hog all the machine's memory (Quill), they run riot and write all over the display even if you have switched away from that program (Vision Mixer or some clock programs), the programs are impure (they modify their own code to some degree like Turbo programs, making it impossible for one copy of the program code to run as several incarnations of itself), or they resize their windows or do something to require a 'guardian' window. The hotkey systems provides some variations on the standard commands described in the last issue to help tame these eccentric behaviours.

The special versions add a single letter parameter at the end of a HOT_LOAD, HOT_RES or HOT_CHP definition to specify the 'special actions' needed for this particular badly behaved program.

P - Protect memory from a program like Quill
I - impure program (modifies its own code)
F - Freeze job when buried
G - Guardian window
U - Unlockable window

Some will require additional parameters such as Guardian Window sizes or amount of memory.

When Quill starts, it looks to see how much free memory it can find. This could well be most of the free memory in the machine, even if your little document only needs a few kilobytes. In the past, some workarounds such as Simon Goodwin's DIY Toolkit Taskforce program have used such tricks as dimensioning large arrays to expand SuperBASIC to prevent Quill getting too much memory, then releasing that memory by clearing the array. There is a simpler system with hotkeys.

Protecting Memory

By adding a 'P' parameter to the hotkey definitions we can force programs like Quill to be only allowed to take that amount of memory rather than hogging the whole lot to themselves.

```
ERT HOT_LOAD('Q','FLP1_Quill',p,32)
```

This starts a copy of Quill and lets it have a maximum of 32 kilobytes for its files. In practice, it will be slightly less than the amount specified. If you omit the value, you will be asked to specify it - this might be a useful option if you sometimes work with short documents and sometimes with long documents.

```
ERT HOT_LOAD('Q',FLP1_Quill,p)
```

This will ask how much memory Quill is to be allowed. If you are just typing a short letter, you might enter a value of 32, or if you plan to work on a really long document, you might enter a value of 128 or 512.

Impure Programs

If we plan on trying to save memory by making one resident copy of a program run as several copies of itself (e.g. only load one copy of a program running as several copies of itself) we can come unstuck if that program modifies its own code. There aren't many such programs about luckily - the QPAC2 manual implies that the commonest ones are programs compiled with the BCPL, Supercharge and Turbo compilers. There may also be a few QLiberated programs with attached BASIC extensions which modify data storages within their own code - QLiberated programs in themselves don't do this, but are at the mercy of extensions users may include n their programs.

By using the I (for Impure) parameter, we tell the hotkey system that this program may be impure, so it creates a fresh copy of the program for each incarnation run to get around this problem.

Although you can use the I option in a HOT_LOAD command, there is no real purpose to this as by definition, HOT_LOAD simply loads a fresh copy each time it is executed anyhow. It is only really useful in commands which create resident copies of programs, like HOT_CHP and HOT_RES.

```
ERT HOT_RES('U','FLP1_USERS_TURBOD_TASK',i)
```

In practice, unless you are in the habit of making a program compiled with Turbo resident (e.g. that ever so essential business program you wrote and compiled with Turbo) you are unlikely to use this option.

Freezing a Program

There are plenty of old games and a few graphical programs like your editor's old Vision Mixer advertising display program which write continuously to the screen even if you CTRL C out of them for some reason. Programs like that can be tamed with the F parameter, which forces a program to freeze when its windows are buried (CTRL C to another program puts the display of that program theoretically on top of the first program to hide it and so suspend it). The reason given as to why such programs write continuously to the screen rather than legally via the operating system is usually 'speed' - older games poke directly to the screen memory because this can make them write pretty graphics faster than having to undergo the overheads and checks of the operating system calls. Sadly, this means that the program doesn't stop poking to the screen when it should, so the F option forces this to happen.

```
ERT HOT_LOAD('M','FLP1_OLD_GAME',i)
```

Unlockable Windows

There are some programs like on screen clocks and caps lock indicators which need to be able to overwrite certain parts of other programs to do their job as required. The pointer environment does not really like this too much. If you are typing something into a word processor and like to have a multitasking clock display running in a corner and perhaps have a little multitasking program displaying CAPS ON or CAPS OFF at the same time (since the QL keyboard doesn't have a CAPS LOCK indicator), it can be useful to unlock windows to allow them to write all over each other!

```
ERT HOT_LOAD('Q','FLP1_Quill',p,128)
ERT HOT_LOAD('C','FLP1_Caps_task',u)
ERT HOT_LOAD('K','FLP1_Clock_obj',u)
```

The first command allows a copy of Quill to be loaded with 128k of memory for its documents by pressing ALT Q or SHIFT ALT q. The next line allows my caps lock program to be started with unlocked windows so it can overwrite the Quill display when I press CAPS LOCK key. The third line starts my little clock display program (one I wrote myself as I don't like the Toolkit 2 clock).

Guardian Windows

If a program is in the habit of changing the sizes of its own display windows, or needs to access parts of the display its windows don't cover at any time, it can be a bit of a problem as either it may overwrite and damage part of another program's display output or another program may overwrite its display. The pointer environment is not really able to determine from window sizes and locations quite what this program is up to and may get storage of other programs' displays wrong and corrupted. Guardian windows can help here by providing a protected area of the screen. If you are unsure of what size of window to specify (it has to cover all the area accessed by your program) you can specify the full screen of course.

```
ERT HOT_CHP('G','FLP1_NAUGHTY_PROG',G)
```

This defines a guardian window covering the job's area and the full screen.

```
ERT   HOT_RES('G','FLP1_NAUGHTY_PROG',G,
512,128,0,0)
```

which defines an area covering the top half of the QL screen.

Some of these options can be combined, e.g. an impure program may also require a guardian window, or to be frozen. I don't know what combinations are permitted, so experiment a little.

Finally, one additional option allows you to specify strings to be passed to a program when it starts. You may come across some programs which are started like this for example:

```
EXEC FLP1_PROG;'FLP1_'
```

In this case, the string 'flp1_' is passed to PROG. Some programs would use this as a default drive to find startup files for example. In Turbod programs this would appear in the OPTION_CMD$ variable, or CMD$ in QLiberated programs. Filter programs may also make use of this (see Toolkit 2 manual)

Such strings can be passed by adding them at the end of a HOT_CHP or HOT_RES or HOT_LOAD statement, after a semi-colon.

If you have specified the P,F,I, etc parameter options, ensure that this string is passed AFTER these options:

```
ERT    HOT_LOAD('M','FLP1_MYSILLYEDITOR_
task' ,i,g;'FLP1_TEXT')
```

EXEP from the Command Line

Hotkey definitions are useful. It can also be useful to have the extra facilities available as a command in the style of EXEC. The Hotkey system provides an EXEC equivalent called EXEP

```
EXEP FLP1_ARCHIVE
```

works just like EXEC FLP1_ARCHIVE

```
EXEP FLP1_ARCHIVE,P,256
```

starts Archive in 256K of memory

```
EXEP FLP1_MYCLOCK,U
```

starts clock program unlocked

```
EXEP FLP1_GAME,F
```

starts a game which freezes while you CTRL C to another program.

```
EXEP FLP1_GAME,G
```
starts the game with a guardian window covering the whole screen

The throbbing pain in my brain means it's time to stop for this issue. I hope you are finding this series useful. It has been a forced learning experience for me so far to the extent that I find I understand hotkeys now, but still need to refer to instructions as I cannot remember the names of commands, or the options available for each command. This implies to me that the hotkey system is perhaps not too hard to understand, but the quantity and nature of the facilities mean you need to keep a reference guide to hand to make best use of the hotkey system. Equally it is worth noting that if you put the hotkey definitions into a single boot program on the disk you use to start up your system, you don't need to memorise everything, just have the notes to hand if you need to change that boot program from time to time!

You and Your Software - Just good Friends? - Part 6

Geoff Wicks

It is almost midnight. You have just finished typing the 2,000 word document you have promised your boss for tomorrow. Before printing you decide to save it, but have not noticed that the disk is write protected. The computer freezes, and you have to start typing again.

In practice this situation rarely arises. All word processors will warn you that something has gone wrong, and most will tell you precisely what. They will give the chance to correct your mistake. It is something we take for granted. If, however, you make a mistake in printing, your word processor may not be so helpful.

If I attempt to print from the QL when my printer is set up to print from my PC, no QL word processor tells me what has gone wrong.

User friendly programs are "error trapped". They try to stop your mistakes or their own internal shortcomings from causing a fatal crash on your computer. If a fatal crash does occur, they will tell you what has gone wrong.

QL error trapping has received little attention in QL publications, probably because it was introduced haphazardly. From the start error trapping was planned for QL Superbasic, but it was not implemented until the JS ROM, and then not effectively. The first trustworthy ROM error trapping came with the Minerva.

In the meantime other forms of error trapping had been introduced. Toolkit 2 contains some commands to trap errors when loading and saving to disk (Section 10.2), and both the Turbo and QLiberator compilers contain their own error trapping. QLiberator even has three different types. Its internal system, the red windows that appear when something goes wrong; general error trapping for use in programs; and support for the error trapping built into the Minerva ROM and SMSQ(-E). The last of these is very useful because if you write a program using Minerva or SMSQ(-E) error trapping and then compile it with QLiberator using one of these systems, the error trapping is also compatible with earlier ROMs.

Just Words! uses a combination of the error trapping contained in Toolkit 2 and the SMSQ error trapping supported by QLiberator. However, my personal opinion is that the best error trapping is contained in the Turbo Toolkit, because it encourages the user to think at two levels. It also has numerous commands to trap disk operation errors.

What do we mean by two levels of error trapping? I sometimes call them "global" and "tailor-made". A similar division would be "non-recoverable" and "recoverable" errors, or "unforeseeable" and "foreseeable" errors. Thinking at two levels encourages us to look at the purpose of error trapping.

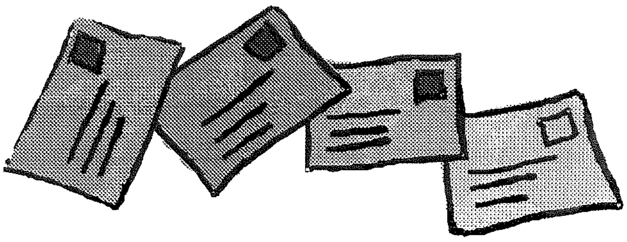
No computer program is perfect. Just Words! programs are relatively simple when compared to say LineDesign or Text87, but they contain 1,000 - 2,000 lines of code. I would be kidding myself if I thought there were no errors and no shortcomings in that code.

This is where global error trapping is important. Error trapping tells me the line where the error, or rather the crash, occurred and the nature of the error. Usually this will not be of much help to the user, because the program will have crashed, but it gives program's author vital information about why and where the program went wrong. A first essential is that error messages make sense.

On my PC the Solitaire game provided with Windows has become corrupt. If I attempt to run the program, I get the error message, "This program has performed an illegal operation and will be shut down." If I ask for details, I am told that "SOL caused a general protection fault in module SOL.EXE at 0001:0000060d", and I am given the advice to contact the "program vendor". How unlike the QL, where you would get a "bad medium" message, and it would be a simple matter to re-copy the corrupted file from the master disk.

Friendly Software Rule:
Make sure your error messages are meaningful to the user.
User friendly software gives the user a clear indication of what has gone wrong when a crash occurs. It should provide him with an idea of what he can do to correct the error, or, if this is not possible, information he can pass on to the author. If, as a user, you have a reason to complain about a program, the worst thing you can do is to say "it doesn't work" or " it was non too successful", as this provides the author with no information about the cause of the problem. Always say where and how the program went wrong, and if possible give details of the error message.
Tailor made error trapping is more difficult to write than global error trapping. It will mainly be needed to cover situations where the error is caused by the user. For example, saving to a write protected disk; attempting to print to a parallel port using "ser"; or inputting a letter when only numerals are permitted. It can include other situations such as carrying out an operation for which the computer has insufficient free memory. In these situations the normal QL error messages are often too superficial, and you need to give the user more guidance about what he can do to recover from the error.
Writing customised error trapping has its own dangers, because the author can make errors in his routines. My program QL-Thesaurus has little used routines for sending the results of a search to a printer or to a disk. These two routines initially share the same error coding, but there is some extra coding for the disk routine to prevent an existing file being overwritten. In an early version of the program I made a mistake in this part of the program, and any user trying to print out the results of a search would have seen the error message, "SER1 already exists. Delete (Y/N)?"
In a QL Today review two years ago, Jonathan Hudson wrote scathingly over a program's reliance on the in built QLiberator error trapping. He said:
"The program should trap data errors and inform the user how to rectify the problem. It is not really acceptable, in a commercial program, to be presented with a low level and incomprehensible QLib error message."
I read this assertion with some ambivalence. Strictly speaking I agree with Jonathan, but I am also aware of the difficulties in writing custom built code. Crash a Just Words! program, and, if it is a recoverable crash, you should get a custom message. If it is non-recoverable, you are likely to get a QLiberator error message.
Next time: Manuals - the great unread.

Letter-Box



Nick Cheesman writes:

I've been a reader of QL Today for a while now so I thought it was about time I wrote to you.
I am a Software Engineer by trade (Visual BASIC mainly) so it seems a logical progression to do some software for the QL. The QL needs new software but being a relative newcomer to the scene, I have no idea what is or isn't already available. There's not much point slaving away on a program only to find no one wants it! Could I therefore invite readers to write in to QL Today and to let all QL developers (not just myself) know what programs you think ought to be written. I look forward to reading the list.
Nick Cheesman njc1@netcomuk.co.uk

David Gilham writes:

Getting email onto your QL without TCP/IP on your QL

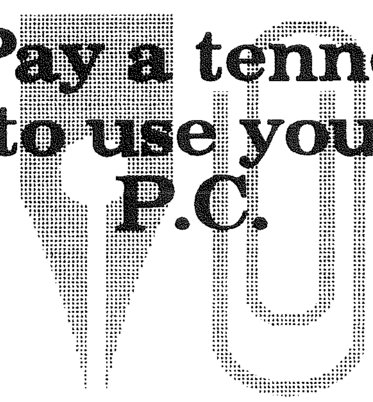
Obviously you must have a decent modem and I would also suggest replacing your 8049 with hermes or Superhermes.You must also have a decent Comms program - I use QTPI which does its job for me. QTPI should be available from most QL PD suppliers and I think a version is in the Quanta library. I stand to be corrected on that point. *[You can certainly download it free from Jonathan Hudson's Web site - Editor]*
Most important for email purposes you must

access via telephone dialup to a system which offers Unix shell accounts or Menu accounts and email. The one I use is Spuddies Xanadu on 01708 442043 in the UK which does not have any charges and offers only email access to the Internet. It is as simple connecting your phoneline to your QL via your modem. Intstalling your Comms program and dialing Spuddy or whoever else has this service available and setting up an account.
The above is a fairly simple description of how to do it for any QL system. For those already into Comms it might seem a bit verbose but it is as simple as the above. On Spuddy you do not get anything like ftp or web browsing, just a simple email service and a minimal selection of Usenet newsgroups. Unfortunatly he has only two lines in for his users.
If anyone has any info on any other systems offering a similar service which can used by the QL community please put a note in Quanta or QL Today. I do not want to overwhelm spuddy too much.
qluser@spuddy.mew.co.uk

Mr. Tanner writes:

The latest Byts of Wood prompts a comment. Then what's so unusual about that?
R.W. is looking for a server which will allow a QXL user to access files held in the PC host's directories.
There are barriers to this, especially to the computing rabbits among whom I count myself. But what a rabbit may do is to dig a burrow ("pipe", or perhaps "thing", in posh computerspeak) under the obstacle.
For a number of years, now, I have been passing files to and fro' between the programs in one of my QXLs and its host '286.
This is quite readily done via win2_, aka d:\qxl.win. Since d: is a virtual device, this volume constitutes an area of extended memory in the PC which is accessible both to DOS and SMSQ. Also, since qxl.win is a DOS file it may be treated as such. Although it is seen by SMSQ as having been formatted to 1Mb, its actual physical size is originally a contiguous 134kb.
It contains two QDOS files. The first is of length 128kb, which is never changed (well SMSQ never sees it as having been changed), and which is used for the bi-directional exchange of object files, and ASCII down. The second is used for transferring ASCII files up from the QXL. Its length is initially 2kb, and its data blocks sit at the end of the physical file, so that if it is required to be extended, then the necessary additional sectors are provided by DOS, and are seen as logically contiguous by my programs.
For object file transfers from QXL to PC, tranches of memory are FS.SAVEd into the 128k buffer, which on the DOS side my PASCAL programs extract by direct memory addressing. And SMSQ simply writes out ASCII files which are then handled as conventionally blocked DOS binary files. In the reverse direction both types of file are written into the buffer by d.m.a., and retrieved as QDOS files by SMSQ.
These arrangements have worked well for some years. (They could have performed even better were it not for this slave block nonsense, however don't let's get diverted up that particular dead end). But they are tied to a specific set of programs. I see no reason why a proper practitioner should not be able to write a pair of general purpose handshaking programs, using similar principles, which would constitute a limited form of file server.
There is, of course, no reason why the PC on which the DOS filestore is located should also be host to the QXL. In fact, should I finally fail in my current attempts to be supplied with a working successor to my dear old QL, it is possible that I may have to fall back on an arrangement where my two QXLs are in separate PCs, each accessing the DOS directory of the other's host. I am quite sure that Mr. Barker's excellent QL-PC server will do all that is required for this.
R.W. also seems to be under the impression that the reason for many people having drifted off into using a PC is the "information interchange factor". I fear that it may be more fundamental than that.
My own needs are two fold: for computing, where the QL's successors are still ahead of anything else that I might afford, and for the sort of non-computing jobs of which any old machine (AOM) might be capable. These include source editing, spreadsheeting, word processing, and the like.
For a number of years I ran the one QXL, which did the computing, as a slave from a master QL. The former's host PC performed the AOM's tasks.
When I acquired my second QXL I had anticipated that at least some of this humdrum workload could be passed across to the first one, to my advantage.
I was astonished to discover that this would not be the case. In fact it transpired that AOM was better at doing the things that AOM can do than QXL/SMSQ was. And very much better than when the latter was confined into the PE straitjacket.
(Mind you, AOM shows up to even greater advantage when compared with WINDOWS-infected machinery, but that is another matter).

This is illustrated by the truly dreadful (dreadful, that is, to an AOM user) fankle R.W. seems to have got into over line terminations.
Both my working editors accept files with either "PC" or "UNIX" terminations, i.e. with or without the ‹CR› as did my former, and as does my present word processor. In fact they will quite cheerfully merge files with differing terminations. Files so loaded are neutral, so that there is no need for a file format window.
When they come to be written out, either by my editors or by my older w.p. in "editor mode", the choice of termination is left to the user. This can be done on an ad hoc basis for a particular file, or saved as an option. Saving may either be local, to a particular session, or global, retained in a configuration file. Any number of these last can be maintained, and specified as required in the start-up command line.
In addition one of my editors, that which supports folding, allows the choice of termination to be made extension dependent. Again, this can be held in a configuration file, which is created/updated either from within an editing session, or as a separate external operation.
The above is common practice among AOM utilities.
Contrast with R.W.'s experience. He airs his difficulty. Nanny comes running, and soothes his wails by showing that she has in fact already attached an appropriate bell to his perambulator. Suppose she had not? Suppose he would really have preferred a whistle to a bell?
If I should complain of a similar problem in AOM, I would expect to be told to get off my butt, and set up the necessary default line in the relevant configuration file. Serve me right, too.
The difference is between a prescriptive environment, and an enabling one within which the user is free to make his/her own mistakes. Which last opportunity I exploit to the full, believe me.
None of the above will be liked. But then I have just recently embarked on my forty-first year of involvement with digital computing, and in every one of the previous forty I have succeeded in getting up the noses of real, non-quiche-eating, computer people. I see no reason why I should change now.
■



Pay a tenner to use your P.C.

At JUST WORDS! we are not paranoid about the PC. It is there and many of us have to use it daily. We have therefore written the most versatile possible utility to transfer files between your QL and PC word processors. Your QL word processor can be Quill, Perfection or Text87, and your PC word processor either Windows or Dos based. The features of this pointer driven program include:

Correct transfer of accented and similar characters (of special interest to non-english language QL users).

Possible transfer of bold, underlined and italic text (depending on your word processing software).

Setting font, font size, justification, margins and tabs on the QL for windows based word processors.

Experimental e-mail reading and OCR-text tidy routines.

QL-2-PC TRANSFER costs just £10 or 15 Euros. Its is available either directly from JUST WORDS! or from QBRANCH.

Payment can be made by sterling cheque or giro transfer in Euros to Netherlands Postbank number 4111942 (G.T. Wicks).

Geoff Wicks, 28 Ravensdale, Basildon, Essex SS16 5HU, United Kingdom.

Telephone: +44 (0)1268 - 281826

e-mail: geoffwicks@hotmail.com

Q40 News

Q40 News from Claus Graf

There has been the question on the QL mailing list of converting sound files on the Q40. A few days later there is the tool to do that. It is called sox. This has been ported to QDOS by Jonathan Hudson. So sometimes there is some delight on the Software side of the QL! You can get sox from

www.bigfoot.com/~jrhudson/qdos/qss-sox.zip

Sox knows of a lot of sound formats, and it can convert to ub (the Q40 sound file format). Here is an example command line that converts wav-Files to ub (ub can be played with qsplayer, see www.q40.de):

```
EXsox;"input.wav-c2-r20000-tuboutput.ub
resample"
```

Remarks: -c2 -› two channels, -r20000 -› 20kHz, -tub -› unsigned byte format (for Q40), resample -› better quality.

Setting up the Q40 Sound System

Tim Swenson

One of the neat hardware features of the Q40 is the built in Sampled Sound System. As much as the hardware has been there, only recently has some software become available that makes the sound system available to the average user. This software, Qsplayer, sends a sound file to the sound system, which then outputs the sounds to the speakers.

The Sampled Sound System is fairly simple. It just takes two bytes (one for the left speaker and one for the right speaker) and sends them out the the speakers. The tough part is getting the right bytes sent to the speakers to make a proper sound. There is no SBASIC commands to generate the right bytes to create sounds and tones. I don't believe any C68 libraries have been created. At the moment, the only way to use the Sound System is to play existing sound files.

Now, this may seem a limitation, but for a QDOS system, this is entirely new. The sound file can be music, sounds from a TV show or movie, or what ever you can find. I am sure any Q40 user will be amazed when they hear full stereo sound coming out of the Q40.

Setting Up the Hardware

The Q40 has two different connectors for the Sound System. They are Line Out and Speaker Out. Line Out is for connecting to headphones or amplified PC-type speakers. Speaker Out is for connecting to the small speaker mounted in the PC case. Both connectors put out the same sound signal, but I don't know if there are any electrical differences between the two (meaning that you could hook up the Speaker Out to an amplified speaker). The hardware differences are that Line Out has three lines (Ground, Left, and Right) where as Speaker Out has four lines (Left, Ground, Right, and Ground).

The speaker on the PC case is capable of being used for the Q40 Sound System, but it is going to be very weak and only in mono (it's only a small little speaker). Using an amplified PC-type speaker, you can get stereo and volume control.

Since I already had Speaker Out hooked to the PC case speaker, I decided to use Line Out for the PC-type speakers. The materials I needed are:

- 3-pin jumper cable
- 1/8" (3.5mm) female phone jack connector
- PC-type speakers

The 3-pin cable was kind of hard to find. I did not find it at my local Radio Shack, but had to trek down the the Silicon Valley Mecca-store, Fry's. Fry's started out selling computer parts to the computer geeks of Silicon Valley. They now carry appliances, phones, etc, but they still have a good section of electronic parts. The particular cable I picked out was labelled as "CPU Cooling Fan Wire Extension". All I needed was a 3 wire cable that has the jumper connector to fit over the Line Out jumper pins. I've had some discussion with Tony Firshman about using a shielding wire. The only problem with not using shielding is that the Q40 may cause some noise when using the speaker. Since I found out about the shielding after I built my cable, I'll have to live with what ever noise there might be.

The 1/8" (3.mm) female phone jack connector was where the PC-type speaker was boing to plug into. Any type of the connectory will work, I just picked an open circuit type because it was cheap. For those that have access to Radio Shack, the part number is 274-249A.

The PC-Type speakers are just the generic speakers that you can find down at the local computer shop. I went to my local one and picked out the cheapest pair that I could find ($8 US). You

will get a left and right speaker, with one of them having a on/off switch and volume knob. Mine had a 3D on/off button. All it seems to do is boost the output a bit.
To build the cable, I just snipped off the one end of the 3-pin cable (the part that will not fit onto the motherboard). I then soldered the bare wire into the phone jack connecting, making sure that I know which wire was Left, Right and Ground. The cable is long enough for me to run the connector out of the case through one of the 9-pin D connector holes. I did not attach it to anything, but just left it hanging. I don't plan on plugging the speaker in and out very much.
Now I just plugged the speakers into the jack and into a power outlet. The hardware side was now done.

Setting Up The Software

Now that the hardware is setup, time to work on the software. Available on the Q40 website (www.Q40.de) is Qsplayer. This program takes a sound file (_ub) and sends it to the sound system. The program comes in command-line and Pointer Environment versions. I have only tried the PE version so I can't say how the command-line version works.
Upon unzipping the Qsplayer archive file, there is the Qsplayer executable, a number of example sound files, and a README file for the sounds. There is no documentation on how to use Qsplayer. Since documentation is the last thing I read anyhow, I just executed Qsplayer. A typical PE window pops up with the usual buttons on top of the window, plus a Play and Stop button in the lower left part of the window.
I clicked on File, a file section window popped up (similar to QMENU File Select window), and I browsed around the file system until I found the _ub files. I selected one and did a DO on it (right mouse button). I was then back to the main window and I heard the sound coming from the speakers (Oh, I did make sure the speakers were hooked up and turned on). Hitting the Play button makes the sound play over again.
That's all there really is to playing a sound on the Q40. The next problem is going about getting sounds to play. Since there does not seen to be a Line In port on the Q40, it is not possible to have the Q40 sample the sound from another source. So, the next thing to do is to figure out how to convert sound files in other formats (.wav, .au) to _ub format.
As luck would have it, Jonathan Hudson came to the rescue. Jonathan ported a unix program called 'sox' that converts sound files between different sound formats. Jonathan included the support of the _ub sound file.
The sox zip file is available off of Jonathan's web page. It includes the full source, including two programs that convert two different modem sound files, so it's a good size download. For the general user, the only files that will be of interest are two README files, explaining how sox works, and the sox executable itself.
So, to test it out, I downloaded sox, unzipped it to RAM, and copied just those files I needed to disk. I also downloaded a few .wav files from a couple of different WAV web pages. One of the README files tells me the exact command line I need to run sox and have it convert a .wav file to _ub.

```
exec sox;"file.wav -c2 -r20000 -t.ub file_ub"
```

The only thing that changes when you run sox is the name of the input and output files, file.wav and file_ub respectively.
Sox fires up and opens the usual C68 window, but does not display anything in the window. I didn't panic, because sox works silently. Depending on how large the .wav file is, sox can take a minute or two to run. So I gave it some time and the C68 window soon disappeared and the conversion is complete. I than ran Qsplayer and tried out the new sound file.

The End Result

To test the quality of the sound generated on the Q40, I played the .wav files on my PC first, then played them on the Q40. Except for the differences in the speakers between the two systems, I could hear no difference in the quality of the sounds.
Now I can spend hours downloading .wav files off the net just to hear some bald guy to "D'oh".

Q40 Colour Driver

Dave Westbury

In a recent version of SMSQ/E for the Q40 appeared the long awaited extended colour driver. This allows access to the full graphics capabilities of the Q40 i.e., 1024 by 512 pixels with each individual pixel colour from a palette of 65536 (fixed) colours. Wonderful! but there is of course one slight drawback with such luxury. Whereas the old QL screen gave us four or eight colours at up to 512x256 it only took 32K memory. With the Aurora at the highest mode 4/8 setting (1024x768) this is increased to 192K. But now at 1024x512 the Q40 system has up to 1Meg of display memory to shift about (e.g. scrolling the screen). Just as well we have a fast 40MHz 68040 with 32bit display access.

Unlike the Aurora on the Q40 there is no trade off between display resolution and available colours so that when running in the hi-colour hardware mode it doesn't make any difference between MODE 4, 8 or say 65536. This means that theoretically you could have MODE 4 and MODE 8 jobs running alongside each other. The reality, at least at present though, is you can't; the MODE command now doesn't have any effect on the display mode (but you can read the high mode setting via TRAP#1). This prevents MODE 8 jobs from being displayed properly - I suspect that any support for MODE 8 jobs will require some clever workaround, e.g. the screen driver will need to map to double width size pixels.

There are a number of new SBASIC commands for controlling colour. Instead of just simply extending the colour parameter range you would normally give in a PAPER, INK, etc., command it allows you set colour schemes for each SBASIC job. This is very useful since it not only allows existing SBASIC programs to work as normal without any changes (except MODE 8 jobs for the reason mentioned above), but it allows the standard QL colours to be mapped to any of the different colours available on your hardware.

`COLOUR_QL`	selects standard QL colour definitions (default).
`COLOUR_PAL`	selects a 256 colour palette mapped definition.
`COLOUR_24`	selects true colour (24 bit) definition.
`COLOUR_NATIVE`	selects native H/W colour definition.

The first two schemes can have the colour palette maps redefined via:

```
PALETTE_QL start, new colour for QL value n,
n+1 ...
PALETTE_8 start, new colour for PAL value n,
n+1 ...
```

The 'new colour' value is that from the true colour range (24 bit = 0 to $FFFFFF). On the Q40, since it hasn't got a hardware palette map, any new colour assignments only affect new drawings. On machines with true palette maps (e.g., PC's) apparently this would have an immediate affect on any information already drawn on the screen.
Also available from SBASIC is the ability to set a 'wallpaper' picture background or a plain/stippled colour background. This replaces the normal black screen which would otherwise show wherever there is no job window on the screen display. This wallpaper/background does not appear as an open SCR/CON channel in the system, to revert back to normal you would have to reset it to a black background.

`BGIMAGE filename`	(i.e., a 'win1_wallp' screenshot)
`BGCOLOUR_QL colour`	set background to QL colour (0-255)
`BGCOLOUR_24 true colour`	set background to plain true colour

For SBASIC that is, err, basically it. It is very straightforward to use the new colours in your own programs but it still allows existing programs to run without any need for changes. Perhaps the only thing that may be encountered is if a program written for MODE 4 colours makes an assumption (or inadvertent) that colour 1 (blue) is same as 0 (black) or 2=3, 4=5, 6=7. This would result in odd MODE 8 colours appearing, but this can be easily solved neatly with a
`PALETTE_QL 0,0,0,2,2,4,4,7,7`
statement.

Briefly for techies, operating system call access to the new colour driver is via new Trap #3 calls: colours (D0=$50 to $5B), blocks ($5C to $5E), palette control ($60 & $61) and wallpaper setting ($6B). Unlike SBASIC all schemes can be used simultaneously to define a colour (QL, palette, true or native). The (very few) modifications made to the internal data structures due to the new extended colour driver has imposed some limitations on colour depth: 16 bit stippled colours, 32 bit plain colours, 8 bit stippled borders and 16 bit plain borders. Much more detailed information is contained in the document 'Graphic Device Interface Version 2' by Tony Tebby, which interestingly also makes mention of a GD3 driver.

Gee Graphics! (On the QL?) - part 14

Herb Schaaf

Carpenters, Chaiken, Catmull & Clark;
Cutting corners from conferences - Around the bend(s) into the year 2000.

If you have a number of sharp corners and want to make them into a sequence of smooth curves, how would you do it?

I've amused myself with a couple of ideas that were introduced into computer graphics decades ago. I ran across them in a book of papers given at a conference at the University of Manchester in 1984[1]. At a 1974 conference in Utah, Chaiken presented a computer algorithm for a method that had been used by carpenters (and probably stonecutters too?) over the years to round off the sharp corners on their handiworks.[2] In 1974 Catmull did his thesis at Utah.[3] In 1978 Catmull and Clark[4] published an algorithm that gives pleasing parabolic arcs.

Consider a zig-zagging collection of connected lines with their vertices. Somehow we divide the straight segments into shorter segments and also create new vertices in-between. We now have another line with less zig-zag, more and shorter segments, and more vertices. The hope is to do this in such a way that the short segments begin to look more like a smooth curve.

Chaiken's use of the carpenter's approach simply cuts off the corners, the question being how much of the corner gets cut off and how much is left in place for the next cut(s), if wanted. As usual, it can be a matter of judgement as to what is pleasing, and when to stop. The ratio of corner removed to corner remaining can vary from 0 to 1/2, where 0 means no cutting and 1/2 means cut them all off. Another way of expressing the ratio is as a unit fraction such as one-third, one-quarter, etc. The general rule-of-thumb used by carpenters was usually one-quarter. With this ratio of 0.25 the smooth curve approaches a sequence of parabolae, also described as a quadratic B-spline. Revealing yet again more regions of my ignorance (also known as opportunities for discovery and learning).**

In the listing Connect_dots_bas when you choose Chaiken's approach you are invited to try changing the ratio to see the effect. You are also invited to decide how many points are enough for a 'smooth' curve. The default values will be used if you simply touch the ENTER key.

Catmull & Clark's approach uses a bit of influence from the cut-off vertex and results in a somewhat similar curve. It also makes frequent use of a mid-point function to recursively generate a cubic B-spline curve.

When you choose the Catmull-Clark option you are invited to change the weighting factor for the vertex. You can decide how 'smooth" by choosing the maximum gap between pixels. As before, defaults are used if you simply touch the ENTER key.

As a final exercise I decided to produce something for the New Year, New Century, and New (are we there yet?) Millennium.

Choose Y 2 K and see what you get!

References:

1 The Mathematics of Surfaces, J.A. Gregory, ed. 1986 Clarendon, Oxford University Press.

2 G. Chaiken, An Algorithm for High Speed Curve Generation. Computer Graphics and Image Processing, vol.3, p. 346-349, 1974.

3 E. E. Catmull, A subdivision algorithm for computer display of curved surfaces., PhD Thesis, Department of Computer Science, University of Utah, December 1974.

4 E. Catmull and J. Clark, Recursively generated B-spline surfaces on arbitrary topological meshes. CAD vol. 10 p. 350-355, 1978.

** Add B-splines, Bezier splines, cubic splines, quaternions, Midi, Clifford algebras, matrices, vectors, tensors, spinors, wavelets, surflets, and stochastic imaging to the growing list of things I'd like to know and understand. But I'd best get back to 3D transforms as promised in the previous century. Next time?

```
100 REMark Connect_dots_bas for GG#14 HL Schaaf Dec 24, 1999
110 squish_x = 1 : IF VER$ = "JSU" THEN squish_x = .85
120 WTV : pixel_height = 101/202 : explain_menu
130 :
140 REMark zig-zag line
150 DATA 6, 10,10, 30,90, 60,50, 100,50, 125,10, 150,90
160 :
170 DEFine PROCedure Chaiken
180  MODE 4 : PAPER 0 : INK 4 : CLS
190  CSIZE#0,1,0 : INK#0,7 : CLS#0
200  raise_up = 0 : move_over = 0
```

```
210  REPeat get_ratio
220   ratio$ = ".25"
230   PRINT#0; "please ENTER a ratio from 0 to 1/2"
240   PRINT#0; "the default value is .25"
250   INPUT#0;' '; ratio$
260   IF ratio$ = '' THEN
270    ratio = .25
280   ELSE
290    ratio = ratio$
300   END IF
310   IF ratio >= 0 AND ratio <= .5 : EXIT get_ratio
320  END REPeat get_ratio
330  REPeat get_point_limit
340   limit$ = "100"
350   PRINT#0; "please ENTER a limit for the final number of points"
360   PRINT#0; "the default value is 100"
370   INPUT#0;' '; limit$
380   IF limit$ = '' THEN
390    limit = 100
400   ELSE
410    limit = limit$
420   END IF
430   IF limit > 0 : EXIT get_point_limit
440  END REPeat get_point_limit
450  counter = 0 : round = 0 : RESTORE 150
460  get_points : draw_lines pts : INK 7
470  REPeat Chaiken_curve
480   IF counter : INK 0 : draw_lines pts
490   DIM tmp(2*(num_points+1),2)
500   counter = 1 : round = round + 1
510   edges = num_points-1
520   tmp(1,1) = pts(1,1) : tmp(1,2) = pts(1,2)
530   FOR edge = 1 TO num_points -  2
540    counter = counter + 1
550    tmp(counter,1)=longx(edge)
560    tmp(counter,2)=longy(edge)
570    counter = counter + 1
580    tmp(counter,1)=shortx(edge+1)
590    tmp(counter,2)=shorty(edge+1)
600   END FOR edge
610   counter = counter + 1
620   tmp(counter,1)=pts(num_points,1)
630   tmp(counter,2)=pts(num_points,2)
640   num_points = counter
650   DIM pts(num_points,2)
660   FOR i = 1 TO counter
670    FOR j = 1 TO 2
680     pts(i,j) = tmp(i,j)
690    END FOR j
700   END FOR i
710   INK 7 : draw_lines pts : CLS #0
720   PRINT #0;,"Chaiken corner cutting round ";round;
730   PRINT #0;" with ";num_points;" points"
740   IF num_points > limit : EXIT Chaiken_curve
750   PRINT #0\\,,"touch [space bar] for another round "
760   PAUSE
770  END REPeat Chaiken_curve
780  PRINT #0\\,, num_points;" points exceeds limit of ";limit
790  PRINT #0,,,"touch [space bar] for menu"
800  PAUSE : explain_menu
810 END DEFine Chaiken
820 :
830 DEFine FuNction longx(edge)
840  RETurn pts(edge,1)+((pts(edge+1,1)-pts(edge,1))*(1-ratio))
850 END DEFine
860 DEFine FuNction longy(edge)
870  RETurn pts(edge,2)+((pts(edge+1,2)-pts(edge,2))*(1-ratio))
880 END DEFine
890 DEFine FuNction shortx(edge)
900  RETurn pts(edge,1)+((pts(edge+1,1)-pts(edge,1))*(ratio))
910 END DEFine
920 DEFine FuNction shorty(edge)
930  RETurn pts(edge,2)+((pts(edge+1,2)-pts(edge,2))*(ratio))
940 END DEFine
950 :
960 DEFine PROCedure Catmull
970  PAPER 0 : MODE 4: INK 4 : CLS
```

```
980  INK#0,7 : CSIZE#0,1,0 : CLS#0
990  raise_up = 0 : move_over = 0
1000  REPeat get_v_weight
1010   v_wt$ = ".5"
1020   PRINT#0; "please ENTER a weight for the vertices"
1030   PRINT#0; "from 0 to 1.  The default value is .5"
1040   INPUT#0;' '; v_wt$
1050   IF v_wt$ = '' THEN
1060    v_wt = .5
1070   ELSE
1080    v_wt = v_wt$
1090   END IF
1100   IF (v_wt >= 0) AND (v_wt <= 1 )  : EXIT get_v_weight
1110  END REPeat get_v_weight
1120  REPeat get_pix_gap
1130   max_pix_gap$ = "15"
1140   PRINT#0; "please ENTER a maximum pixel to pixel gap"
1150   PRINT#0; "the default value is 15"
1160   INPUT#0;' '; max_pix_gap$
1170   IF max_pix_gap$ = '' THEN
1180    max_pix_gap = 15
1190   ELSE
1200    max_pix_gap = max_pix_gap$
1210   END IF
1220   IF max_pix_gap > 0 : EXIT get_pix_gap
1230  END REPeat get_pix_gap
1240  round = 0 : RESTORE 150 : get_points : draw_lines pts
1250  Catmull_Clark : show_status
1260  AT #0,2,8 :PRINT #0;" touch [space bar] for menu        "
1270  PAUSE : explain_menu
1280 END DEFine Catmull
1290 :
1300 DEFine PROCedure Catmull_Clark
1310  REPeat catmull_curve
1320   DIM tmp((2*num_points)-1,2) : round = round + 1
1330   vertices = num_points-2
1340   edges = num_points -1
1350   FOR edge = 1 TO num_points-1
1360    tmp(edge*2,1)= mid (pts(edge,1), pts(1+edge,1))
1370    tmp(edge*2,2)= mid (pts(edge,2), pts(1+edge,2))
1380   END FOR edge
1390   FOR vertex = 1 TO num_points
1400    tmp((2*vertex)-1,1)=pts(vertex,1)
1410    tmp((2*vertex)-1,2)=pts(vertex,2)
1420   END FOR vertex
1430   FOR points = 3 TO DIMN(tmp)-1 STEP 2
1440    mid_x = mid(tmp(points-1,1),tmp(points+1,1))
1450    mid_y = mid(tmp(points-1,2),tmp(points+1,2))
1460    tmp(points,1) = mid_x - (v_wt * (mid_x - tmp(points,1)))
1470    tmp(points,2) = mid_y - (v_wt * (mid_y - tmp(points,2)))
1480   END FOR points
1490   tmp((2*num_points)-1,1)=pts(num_points,1)
1500   tmp((2*num_points)-1,2)=pts(num_points,2)
1510   max_gap = 0
1520   FOR i = 2 TO 2*num_points-1
1530    gap = dist_btwn(tmp(i-1,1),tmp(i-1,2),tmp(i,1),tmp(i,2))
1540    IF gap > max_gap  : max_gap = gap
1550   END FOR i
1570   num_points = (2*num_points)-1
1580   DIM pts(num_points,2)
1590   FOR i = 1 TO num_points
1600    FOR j = 1 TO 2
1610     pts(i,j) = tmp(i,j)
1620    END FOR j
1630   END FOR i
1635   IF max_gap <= (max_pix_gap * pixel_height) : EXIT catmull_curve
1640   IF choice = 50 THEN
1650    show_status : PRINT #0;, " touch [space bar] for another"
1660    PAUSE : INK 0 : draw_lines pts : CLS#0
1665   END IF
1670  END REPeat catmull_curve
1680 END DEFine Catmull_Clark
1690 :
1700 DEFine PROCedure show_status
1710  INK 7 : draw_lines pts
1720  CLS#0: PRINT#0;," Catmull-Clark round ";round;
1730  PRINT #0;" with ";num_points;" points,"
```

```
1740  PRINT #0;," none more than ";max_gap/pixel_height;" pixels apart"
1750 END DEFine show_status
1760 :
1770 REMark two
1780 DATA 16, 5,40, 3,45, 3,50, 5,55, 15,60, 30,60, 37,45
1790 DATA 20,25, 10,20, 0,5, 0,0, 0,0, 0,0, 25,0, 30,0, 35,5
1800 REMark zero
1810 DATA 9, 0,30, 0,60, 15,60, 30,60, 30,30, 30,0, 15,0, 0,0, 0,30
1820 :
1830 DEFine PROCedure MM
1840  PAPER 0 : MODE 8: INK 4
1850  SCALE 100,0,0 :  CLS
1860  CSIZE #0,3,1: FLASH #0,1 : CLS#0
1870  PRINT #0,"Working on it - Please wait"
1880  max_pix_gap = 5 : raise_up = 25 : v_wt = .5
1890  RESTORE 1780 : move_over = 5
1900  get_points : draw_figures
1910  RESTORE 1810 : move_over = 48
1920  get_points : draw_figures
1930  FOR digits = 3, 4
1940   FOR i = 1 TO DIMN(pts,1)
1950    pts(i,1) = pts(i,1)+(40*squish_x)
1960   END FOR i
1970   draw_lines pts : dot_spot pts
1980  END FOR digits
1990  FLASH#0 ,0 : INK#0, 7 :CLS #0
2000  PRINT #0," Happy  New  Millennium  !!"
2010  PAUSE : explain_menu
2020 END DEFine MM
2030 :
2040 DEFine PROCedure get_points
2050  READ num_points
2060  DIM pts(num_points,2)
2070  FOR i = 1 TO num_points
2080   FOR j  = 1 TO 2
2090    READ pts(i,j)
2100   END FOR j
2110   pts(i,1) = (pts(i,1) + move_over) * squish_x
2120   pts(i,2) = pts(i,2) + raise_up
2130  END FOR i
2140 END DEFine get_points
2150 :
2160 DEFine PROCedure draw_points(array)
2170  FOR i = 1 TO num_points
2180   POINT array(i,1),array(i,2)
2190  END FOR i
2200 END DEFine draw_points
2210 :
2220 DEFine PROCedure draw_lines (array)
2230  POINT array(1,1),array(1,2)
2240  FOR i = 2 TO DIMN(array)
2250   LINE TO array(i,1),array(i,2)
2260  END FOR i
2270 END DEFine draw_lines
2280 :
2290 DEFine PROCedure draw_figures
2300  Catmull_Clark : INK 7
2310  draw_lines pts : dot_spot pts
2320 END DEFine draw_figures
2330 :
2340 DEFine  PROCedure  dot_spot(array)
2350  FOR spot = 7 TO 2 STEP -1
2360   INK spot
2370   FOR i = 2 TO DIMN(array)
2380    FILL 1
2390    CIRCLE array(i,1),array(i,2),spot/2
2400    FILL 0
2410   END FOR i
2420  END FOR spot
2430  INK 0 : draw_lines array
2440 END DEFine dot_spot
2450 :
2460 DEFine FuNction mid (n1,n2)
2470  RETurn ( n1 + n2 )/2
2480 END DEFine
2490 :
2500 DEFine FuNction dist_btwn(xpt,ypt,x,y)
```

```
2510  xdis=(x-xpt) :ydis=(y-ypt)
2520  sqdist=((xdis*xdis)+(ydis*ydis))
2530  dbtw=0
2540  IF (sqdist):dbtw=SQRT(sqdist)
2550  RETurn dbtw
2560  RETurn xdis
2570  RETurn ydis
2580 END DEFine
2590 :
2600 DEFine PROCedure explain_menu
2610  WTV : MODE 4: PAPER 0 : INK 4 : CSIZE 1,0: CLS
2620  PRINT\" Connecting the dots creates a sequence of lines."
2630  PRINT\ " Various methods are used to make a smooth 'curve'"
2640  PRINT " by inserting more dots in the 'right' places. "
2650  PRINT\,"Choose from these demonstrations :"
2660  PRINT\,,1;" - Chaiken's carpenters method"
2670  PRINT\,,2;" - Catmull and Clark's method"
2680  PRINT\,,3;" - Y 2 K  ?"
2690  PRINT\\\,"please choose by touching a number key."
2700  PRINT\\,,,,"ESC to quit"
2710  choice = CODE (INKEY$(-1))
2720  SELect ON choice
2730   = 27 : CLS : STOP
2740   = 49 : Chaiken
2750   = 50 : Catmull
2760   = 51 : MM
2770   = REMAINDER : explain_menu
2780  END SELect
2790 END DEFine explain_menu
2800 :
2810 REMark end listing Connect_dots_bas
```

The function of PROCedures and the procedure for writing FuNctions - 1

Mark Knight

This article discusses the writing of SuperBASIC PROCedures and FuNctions; it is intended to help programmers work out when to use a PROCedure and when to use a FuNction. Some beginners to programming write only PROCedures, thinking that FuNctions contain some complex or incomprehensible mystery; hopefully after reading this such things will be easier to understand. We start by looking at PROCedure writing, taking the very simplest forms and working up to more complicated matters. Once PROCedures have been covered in reasonable detail FuNctions are explained, and lastly some pitfalls to avoid in both.

Why PROCedures? The name's the reason:
Imagine you are writing a program that will be presenting a lot of changing information in a window on the QL screen. Your program has two windows, one used for input only, and the other for output; a frequent task is to clear both windows. One way of doing things (a pretty dreadful way though) is this:

```
100 GO SUB 120
110 STOP
120 REMark screen clear subroutine
130 FOR Chan=0 TO 1
140   PAPER#Chan;0
150   INK#Chan;7
160   CSIZE#Chan;0,0
170   BORDER#Chan;1,2
180   CLS#Chan
190 END FOR Chan
200 RETurn
```

This uses the subroutine facility in SuperBASIC; once this routine is written any part of the program can reset the windows by using GO SUB 120. In Jan Jones original design for SuperBASIC there was no GO SUB or GO TO at all, they were added at the insistence of Sinclair Research to make it easier to convert old BASIC programs to run on the QL: many regard this as a mistake.
GO SUB is poor because the line "GO SUB 120" in a listing tells us nothing about what the subroutine does. Loads of GO SUBs make a complex program very hard to understand unless it is very heavily laced with REMarks. As the program grows and has to be renumbered the GO SUB calls will change their numbers in an unpredictable way; the programmer might remember that GO SUB 120 clears the screen for a while, but if he comes back to alter the listing a year or two after it was written it will be very hard. An equivalent PROCedure can be given a name, like this:

```
100 Reset_WINDOWS
110 :
120 DEFine PROCedure Reset_WINDOWS
130   FOR Chan=0 TO 1
140     PAPER#Chan;0
150     INK#Chan;7
160     CSIZE#Chan;0,0
170     BORDER#Chan;1,2
180     CLS#Chan
190   END FOR Chan
200 END DEFine Reset_WINDOWS
```

This is much better, because even if our program grows into a large and complex one, each time we see "Reset_WINDOWS" in the listing we know exactly what is going on. Even used in this most simple way PROCedures have given us an enormous improvement in the readability of our listing, just by allowing us to replace numbered calls with meaningful names. This improvement in readability makes a program easier to debug by making it easier to understand.
The named PROCedure facility should be used with thought though, otherwise it is little better than subroutine calls. It isn't hard to think up cryptic names which require less typing, but they are little use to another programmer trying to alter your code for their own use. If you come back to a listing several years old to alter it yourself you will also be in a better position if all the PROCedure, FuNction and variable names are meaningful.
As well as the facility to use a name for the PROCedure itself we can also use a variable and ensure that it is local to the PROCedure. Making it local means two things: first it doesn't affect the value of any variable with the same name outside the PROCedure: secondly it will cease to exist when the PROCedure terminates. To understand this try the following:

```
100 Silly=10.01*RND
110 CLS
120 FOR Counter=1 TO 10
130   AT 1,2
140   PRINT "Counter=";Counter,
150   Show_RESULTS
160 END FOR Counter
170 :
180 DEFine PROCedure Show_RESULTS
190   LOCal Counter
200   FOR Counter=1 TO 10
210     AT 1,16
220     PRINT "LOCal Counter=";Counter,
230     AT 2,2
240     PRINT Silly*RND,Silly*RND,
250     PAUSE 20
260   END FOR Counter
270 END DEFine Show_RESULTS
```

When you run it you will understand it a little better. What the LOCal directive does is to instruct the interpreter to grab some memory and make a new variable called "Counter" for use inside "Show_RESULTS". It doesn't matter whether there is already a variable of the same name outside the PROCedure or not, "Show_RESULTS" has its own copy to be used inside that PROCedure and in any further routines called from within it. When the END DEFine is reached or if a RETurn statement is executed all LOCal variables created for use within the current PROCedure are discarded and the memory they used becomes available again.
There is more to LOCal than just setting aside some memory; when using arrays LOCal acts like DIM, both setting aside memory and initialising numeric values in the array to zero. If the array is a string array each string is set to a null string (which is a zero length one). So if you use statements like this at the start of a PROCedure:

```
LOCal TempText$(10,10)
LOCal Temp$(36)
```

...then you need not use DIM within the same PROCedure to set up the array as it is already set up. The string Temp$ will have a maximum length of 36 characters and any attempt to set it longer will be truncated to that 36 character limit. You may also use variables, so:

```
LOCal Temp$(FileNameLength%)
```

...is OK provided the variable FileNameLength% has been set before the PROCedure is called. Make sure that there isn't a LOCal variable within the current PROCedure called FileNameLength% as this can result in serious confusion. Within a PROCedure or FuNction all LOCal directives must be before any other program code except REMarks.

To make clearer the use of LOCal arrays try the following:

```
100 Silly=10.01*RND
110 CLS
120 Do_SILLINESS
130 :
140 DEFine PROCedure Do_SILLINESS
150   LOCal Counter,Column
160   LOCal Rubbish(6,3)
170   FOR Counter=0 TO 6
180     AT 1,16
190     PRINT "LOCal Counter=";Counter
200     Rubbish(Counter,RND*3)=
        RND*Silly*(Counter+1)
210     PAUSE 20
220   END FOR Counter
230    PRINT\\
```

Professional & Graphical Software

ProWesS

ProWesS is a new user environment for the QL. ProWesS is short for "PROGS Window Manager", but it is much more than that. Apart from a new window manager, it contains all the system extensions from PROGS, and is essential if you want to run programs which need these extensions.
The ProWesS reader is a major part of the package. It is a hypertext document browser. This means that text files which include formatting commands (including pictures) and possibly links to other files can be displayed and read in this program. This is used in ProWesS to read (and possibly print) the manuals, and display the help files. The hypertext documents which are used by the ProWesS reader are in HTML format, the format which is popular on Internet to display World Wide Web pages.
Another important aspect of ProWesS is the possibility to allow programs to automatically install themselves on your system, and to be able to run them without resetting the system. This means that, when you get a new program, all you have to do is insert the disk and indicate "start the program in flp1_", a menu option in the "utilities" button. To install a program, you indicate "install software", and the software can be added to your system. This way, you don't need to know how to write a boot file to use the multi-tasking capabilities of your computer.
ProWesS includes many programming libraries. These include syslib, an interface to the operating system, PROforma, a vector graphics system, allowing rendering both on screen and on paper (via a printer driver). The DATAdesign engine is also part of ProWesS. It is a relational database system with a bonus, as you don't even need a key field. You get a powerful record at a time data manipulation extension to the language you already use. Of course it also includes ProWesS itself, the new resolution independent window manager.

PWfile

PWfile is a file management program. It allows you to do all kinds of manipulations on files, like copy, move, rename etc. This allows you to manage your files properly.
To make everything easier, you can choose whether the actions also have to apply to the subdirectories. Also, the program will display how much space the indicated files take up, so that you can know in advance whether there is enough room on the destination device. There are even facilities for copying to DOS disks. You can always limit all selections to file in or excluding certain extensions etc.

PFlist

Easy to use program to create listings on any printer (especially inkjet and laser). This ProWesS application allows you to indicate the files which have to be printed. Each column contains a footer which can include the filename and filedate. The listings always allow perforation. PFlist can create your listings in two columns and in landscape (or both).

fsearch

File search utility with many useful options, like the choice to search only files with a certain extension, and whether or not the directory tree has to be scanned. All occurences of the searchstring will be displayed with line number or offset. You can also use special matching features, like case dependent, matching a space with a stretch of whitespace, and searching for a word dilimited string.

font-utils

manage your font collection. You can preview fonts on screen, see what characters exist in a font and convert Adobe Type 1 and similar fonts for use in ProWesS.

LINEdesign

Create artistic drawings, technical drawings, process bitmaps (even scale and rotate them!), and any kind of vector drawings. You can use grpahics objects to create the most fabulous drawings ever seen. Because LINEdesign is a vector drawing program, any part of the picture can be moved, scaled, rotated, slanted without any loss of precision or resolution. In LINEdesign, pictures are device independant, meaning that the printout will be the same on any printer (e.g. same size and position).
LINEdesign is good at handling text. You can easily put titles and full paragraphs on the page. All the fonts can be displayed at any size, rotation, etc. All the fonts which are available to ProWesS can be used in LINEdesign.
LINEdesign is a drawing program, but it can also be used by people who are not good at drawing. LINEdesign is a great program for making leaflets, posters, and any kind of printed work. Lots of clipart and extra fonts are available from public domain libraries and BBS's. You can even import Adobe Illustrator files.

DATAdesign

Never before has it been so easy to create, fill in and maintain your personal databases. To start a new file, just type the names of the fields. To add or delete a field, no problem, just do it. To change the name of a field, just indicate it. You can choose which fields are displayed and also which records. You can have a hidden comment for each record, look at the file in tabulated form and transfer data to the scrap or hotkey buffer. Files can be memory based (for speed) or disk based (for safety).

Dr. Fr. Hemerijckxlaan 13 /1
2650 Edegem
Belgium

www : http://www.triathlon98.com/PROGS/
email : PROGS@triathlon98.com
tel : +32 (0)3/ 457 84 88 fax : +32 (0)3/ 458 62 07

ProWesS - BEF 2400 DATAdesign - BEF 1200 PWfile - BEF 900 PFlist - BEF 600
LINEdesign - BEF 1200 fontutils - BEF 1200 fsearch - BEF 600

Payment terms :

You have to run ProWesS to make LINEdesign, DATAdesign, fsearch, fontutils and PFlist work (even though DATAdesign uses wman).
All our software is normally supplied on high density (HD) disks. However they can be obtained on double density (DD) disks at an extra costs of BEF 100. To use ProWesS and any of our other packages, you need a system with at least 2MB of memory. You should have a harddisk although a two disk system will also work. The use of SMSQ/E is strongly recommended for optimal use of ProWesS.
If you are VAT registered (specify registration number) or live outside the EEC, the amount to be paid is the total (including postage) divided by 1.21 (no need to pay too much).
Payment can be done by EuroCheque in BEF, or by VISA, EuroCard or MasterCard. Credit card orders can be handled by phone. For credit card, please specify name of card owner, card number and expiry date.

Postage : Costs of postage and packaging have to be added. You can choose the quality. Rate depends on no of programs.

	priority mail			*ordinary mail*		
copies	Belgium	Europe	World	Belgium	Europe	World
one	100	200	240	100	120	145
two	135	340	420	135	190	230
3 or 4	160	560	770	160	310	395
5 to 8	185	870	1250	185	550	705
more	295	1130	1610	295	800	1030

All prices are in BEF, including 21% VAT

```
240   FOR Counter=0 TO 6
250     PRINT
260     FOR Column=0 TO 3
270       PRINT Rubbish(Counter,Column),
280       REMark Note trailing comma in pre-
          vious line is not accidental.
290     END FOR Column
300   END FOR Counter
310 END DEFine Do_SILLINESS
320 :
```

Notice the absence of DIM statements in the program; the LOCal declaration at line 160 makes them unnecessary. Notice also that array index numbers start from zero not from one, so as we have a LOCal array declared as having (6,3) elements this makes seven rows and four columns.
When programming on a QL with a JS ROM or earlier it is important to remember that due to a bug in the interpreter you can't use more than nine LOCal variables in a single PROCedure (or FuNction). Such programs may be loaded and edited without trouble, but if you run them the program will soon crash and the QL may even lock up. Each time LOCal is used at the start of a definition these systems reserve enough memory for just nine new variables in the name table; any attempt to use more results in important system pointers and tables being overwritten so beware.
There is yet another good reason for using LOCal variables; it helps to isolate the internal workings of a PROCedure or FuNction from the rest of the program. If a PROCedure uses only its own LOCal variables it will not interfere with variables belonging to other parts of the program, and so once the PROCedure is debugged it is unlikely to make trouble elsewhere in the program. Debugging a large SuperBASIC program then becomes rather like debugging lots of small programs in the form of PROCedures and FuNctions.

The next thing we do is to introduce parameters to PROCedure writing, and it is here that a great deal more power becomes available.
One way of looking at SuperBASIC and SBASIC defines them as built of five kinds of keywords: there are structure definition keywords (things like FOR, IF, THEN and END, REPeat, DEFine PROCedure etc), variable definition keywords (DIM, LOCal, LET) built-in functions and built-in procedures: the DATA keyword has a category all of its own, as the data definition keyword.
OPEN and PRINT are built-in procedures; neither would be much use if they did not accept parameters; PRINT usually needs something to print (even if it's a blank line) and sometimes a channel number if the item to be printed is not required in the default channel. So we can put a single PRINT instruction on its own to print a blank line, or something like:

```
2350 PRINT#2;"Press a key to continue"
```

...to print the given text to channel 2 (normally the listing window). The "#2" part of this line and the string in quotes are parameters for the PRINT procedure. Parameters can be varied each time a procedure or function is called and so give SuperBASIC much of its power. Instead of the above we could have used variables thus:

```
2350 PRINT#PromptChan%;Message$
```

...and as long as the variables are set up before the code is called we can expect the result to appear in whatever channel the variable "PromptChan%" variable indicates.
Adding parameters to a user-defined SuperBASIC PROCedure is simple enough and adds the kind of facility that the PRINT command has, allowing the PROCedure to operate on any values passed to it by the calling code. Revisiting an old friend of mine, one keyword that would be very handy would be a routine to word-wrap any text string to any window, regardless of window size, CSIZE settings etc. We might want to call it Word_WRAP and have it take two parameters, a channel number and a string.
Let's look at what we want this routine to look like from calling code: it should be called like this:

```
Word_WRAP#1,"Some test text to see if the
routine works properly."
```

...or:

```
Word_WRAP#DisplayChan%,Message$
```

So how do we set up a routine to take parameters? The answer is to set them up in brackets in the line where the PROCedure is defined; then they behave like LOCal variables within that PROCedure. Our word-wrap routine with some demonstration code looks like this:

```
100 WINDOW 252,202,256,23
110 WINDOW#2;252,202,4,23
120 WINDOW#0;504,32,4,224
130 FOR n=0 TO 2
140   INK#n;7
150   PAPER#n;0
160   BORDER#n;1,2
170   CLS#n
180   CSIZE#n;0,0
190 END FOR n
200 MODE 4
210 CSIZE 0,0
220 CSIZE#2;2,0
230 Message$="Some test text to see if the
    routine works properly."
240 Word_WRAP#1,Message$
250 Word_WRAP#2,Message$
260 :
30000 DEFine PROCedure Word_WRAP(Channel%,Any$)
30010    LOCal FirstChr%,SpaceFound%,WrapLoop
```

```
30020   FirstChr%=1
30030   IF LEN(Any$)=0 THEN RETurn
30040   REPeat WrapLoop
30050     IF FirstChr%>LEN(Any$) THEN RETurn
30060     SpaceFound%=" " INSTR
          (Any$(FirstChr% TO LEN(Any$)))
30070     IF SpaceFound%>0 THEN
30080       IF SpaceFound%=1 THEN
30090         FirstChr%=FirstChr%+SpaceFound%
30100         PRINT#Channel%;!"";
30110       ELSE
30120         PRINT#Channel%;!Any$(FirstChr%
              TO FirstChr%+SpaceFound%-2);
30130         FirstChr%=FirstChr%+SpaceFound%
30140       END IF
30150     ELSE
30160       PRINT#Channel%;!Any$(FirstChr%
            TO LEN(Any$));
30170       RETurn
30180     END IF
30190   END REPeat WrapLoop
30200 END DEFine Word_WRAP
30210 :
```

Note that the "#" symbols aren't all needed: I always prefer to prefix channel numbers with a hash symbol to make it clear they are channel numbers, but you don't have to do this with lines that call SuperBASIC PROCedures so lines 240 and 250 could have been:

```
240 Word_WRAP 1,Message$
250 Word_WRAP 2,Message$
```

Inside the SuperBASIC PROCedure note that parameters passed to it may have their own names, as above, and that they may act like LOCal variables, but they may also have other effects. The following may take some time to comprehend but it will reward study as parameters can be more powerful if you understand them well.

Under the SuperBASIC interpreter there are two kinds of parameter passing possible; passing by reference and passing by value. If a parameter is passed by value then changes made to it within the routine make no changes outside that routine; it truly acts like a LOCal variable. If a parameter is passed by reference then any changes to the variable within the routine also affect the variable passed to it as a parameter.

To illustrate this try the following:

```
100 CLS : CLS#2
110 Print_and_Alter 1
120 TestVal = 6
130 PRINT TestVal ! "before."
140 Print_and_Alter TestVal
150 PRINT TestVal ! "After first call and before
    second."
160 Print_and_Alter (TestVal)
170 PRINT TestVal ! "after second call."
180 STOP
190 :
200 DEFine PROCedure Print_and_Alter(AnyNum)
210   PRINT#2;\\AnyNum ! "before changing."
220   AnyNum = AnyNum + RND(1 TO 10)
230   PRINT#2;AnyNum ! "after changing."
240 END DEFine Print_and_Alter
250 :
```

The output from a typical run might be:

```
6 before.
7 After first call and before second.
7 after second call.
```

...in window#1 and:

```
1 before changing. 9 after changing.
6 before changing. 7 after changing.
7 before changing. 15 after changing.
```

...in window#2.

So what does all that mean?

Looking at line 110 we see the call to Print_and_Alter with a parameter of 1; this is known as a value parameter and so it can only be passed by value. The program is passing the value 1 to the routine to use to fill its variable called AnyNum. In line 210 this is printed, then altered in line 220 and printed again in line 230. When the END DEFine is reached the AnyNum variable is thrown away.

In line 130 we pass a variable parameter to Print_and_Alter, and as it is passed alone it is passed by reference. This means that when line 200 is reached the variable AnyNum is set up to be the same as TestVal, so changes to AnyNum inside the PROCedure also change TestVal outside the PROCedure. We say that the AnyNum variable refers to TestVal in this instance.

To prevent passing by reference and cause a variable to be passed by value, simply pass it as an expression. In line 160 we do this by placing the variable in brackets, an alternative would be to use the following instead:

```
160 Print_and_Alter TestVal+0
```

This makes the TestVal parameter into an expression, and in SuperBASIC and SBASIC expressions can only be passed by value. So the value of TestVal is passed over to Print_and_Alter, and the variable AnyNum is set up to stand on its own rather than to refer to TestVal. Changes to AnyNum inside the PROCedure make no change to TestVal in this case.

Most of the time PROCedures and FuNctions make no changes to parameters passed to them so it is easy for a programmer to forget the difference between value and reference parameters. If the difference is the cause of a bug in your program then debugging becomes a most frustrating exercise if you have forgotten. or worse if you never knew.

The Psion PRINTER_DAT Format

Dilwyn Jones

This article summarises the Quill PRINTER_DAT file format for anyone wishing to experiment with it, or to patch individual PRINTER_DAT files where these no longer exist in a given INSTALL_DAT for example. It may also prove useful if you want to write a program of your own which uses PRINTER_dat.

The Xchange_dat files used by Xchange are similar, but allow up to 50 translates I think, although I have not really studied this format. Full information on Xchange is available from Erling Jacobsen's Web site:

In the table below, the codes in the 'TYPE' column are given in the form 'n t' where n is the number of bytes for that code. If not given, assume it means only one byte.

't' has two possible meanings: where given as 'b', it means one single byte only where given as 'c' it implies a character codes string, which consists of one byte signifying the length of the string followed by the characters in the string themselves. If the length byte has a value of 0, it should be interpreted as 'NONE' (i.e. nothing was entered for this code string). If the byte has a value of 255, interpret this as 'DEFAULT' - in other words, the code string is not actually listed here, rather the program uses its own built in default value.

The "hash value" counter below does not include any bytes holding length 255.

SECTION	TYPE	PRESET	NOTES
1	b	"prt1"	Identifier bytes, present for all PRINTER_dats
2	b		"Hash value": sum of lengths of sections 13-33
3	10 b		driver name, padded with spaces to 10 bytes long
If using parallel port or not QL SER1 or SER2:			
4	b	0	0 signifies a parallel port
5	c		port name (usually PAR)
6	n b	spaces	port name padded with spaces to position 34
(no section 7 for PAR option)			
Alternatively, sections 4-6 are laid out like this if using SER1 or SER2			
4	b	1 or 2	Serial port number (i.e. 1 for SER1 or 2 for SER2)
5	b		0 to 4 signifies the parity code 0=NONE, 1=SPACE, 2=MARK, 3=ODD, 4=EVEN
6	2 b		baud rate (75/300/600/1200/2400/4800/9600) The baud rate is stored as a word value (Most Significant Byte first)
7	15 b		spaces to fill up to position 34
8	b		lines per page
9	b		number of characters per line
10	b		paper type (0=cut sheet, 1=continuous paper)
11	4 b		4 spaces - appear to be unused
12	c		code string for end of line codes
13	c		code string for preamble codes
14	c		code string for postamble
15	c		code string for bold on
16	c		code string for bold off
17	c		code string for underline on
18	c		code string for underline off
19	c		code string for subscript on
20	c		code string for subscript off
21	c		code string for superscript on
22	c		code string for superscript off
23	c		code string for translate sequence 1
24	c		code string for translate sequence 2
25	c		code string for translate sequence 3
26	c		code string for translate sequence 4
27	c		code string for translate sequence 5
28	c		code string for translate sequence 6
29	c		code string for translate sequence 7
30	c		code string for translate sequence 8
31	c		code string for translate sequence 9
32	c		code string for translate sequence 10

About Cueshell

Ian Pizer

Albin Hessler, the author describes Cueshell as a Desktop Program for performing everyday tasks (related to files and directories) in an easy way. It works with the Pointer Environment.. It is useable from keyboard but best is with a mouse.

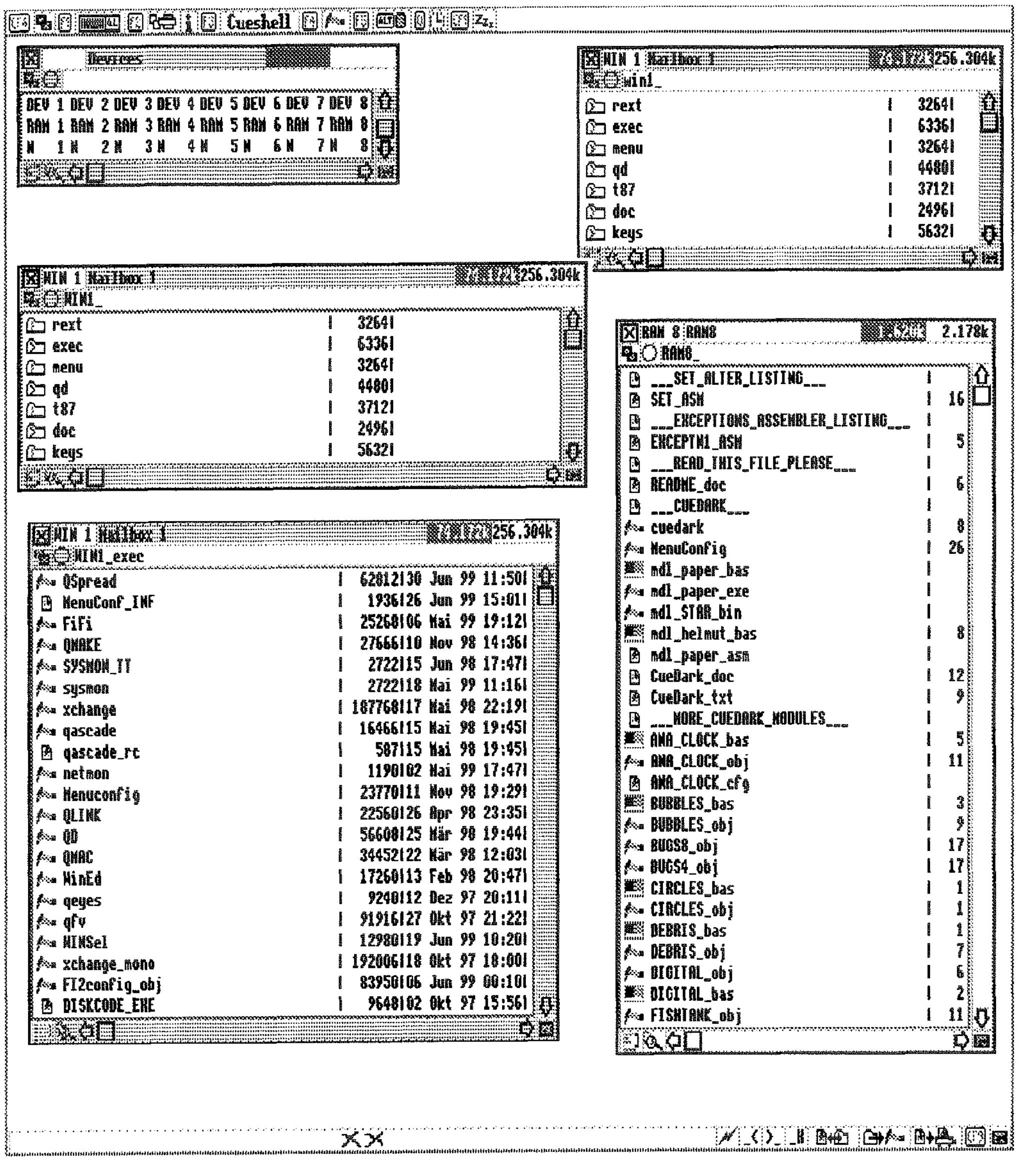
Whenever I have a problem and resort to Cueshell I am delighted how smoothly it solves complex problems. And each time I find a new possibility. If you do not have Cueshell I recommend it to you.

Some weeks ago I lost all the files in win1 but had a backup in win2 so I used Cueshell to copy each sub-directory from win2 to win1 one at a time. That took time and my undivided attention. Then I thought - maybe Cueshell will copy EVERYTHING from one directory to another (be it flp_ win_ or rom_ or whatever). So next time a similar problem to the one mentioned above occurs this is what you can do:
In Cueshell, DO on the good Device name (e.g flp2_). Now you see all the files and sub-directories in flp2_. Hit the O at the top of the list to choose ALL files and sub-directories. Now DO on any point of the chosen area but at the extreme left. This brings up the animated sprite for copying. Move (Drag) the sprite onto the device to which you wish to copy (e.g. flp1_) and hit a mouse key (or space key). You then get a window in which you can choose to Update or Backup. Hit your choice and Cueshell does the transfer of all files and sub-directories and sub_sub directories, with the contained files, for you!

Changing names of files is also very easy with Cueshell.
I have only described one major possibility of the many functions avaliable in Cueshell. Another function I recently found is how to delete a file, a sub directory or even a whole directory. You DO on the chosen item as mentioned above and drag the sprite to the XX symbol at the bottom of the Cueshell window and hit a mouse key and the delete is done.

OK Cueshell is not free but as an enthusiastic user I can highly recommend it. I use Cueshell with AURORA, I presume it works equally well for other QL setups.

CueShell works in any screen resolution - you can resize CueShell itself dynamically, and you can resize any window inside CueShell as you can see in the snapshot to the left.



```
DEV 1 DEV 2 DEV 3 DEV 4 DEV 5 DEV 6 DEV 7 DEV 8
RAM 1 RAM 2 RAM 3 RAM 4 RAM 5 RAM 6 RAM 7 RAM 8
N   1 N   2 N   3 N   4 N   5 N   6 N   7 N   8
```

```
WIN 1 Mailbox 1                     256.304k
win1_
rext                                 |   32641
exec                                 |   63361
menu                                 |   32641
qd                                   |   44801
t87                                  |   37121
doc                                  |   24961
keys                                 |   56321
```

```
WIN 1 Mailbox 1                     256.304k
WIN1_
rext                                 |   32641
exec                                 |   63361
menu                                 |   32641
qd                                   |   44801
t87                                  |   37121
doc                                  |   24961
keys                                 |   56321
```

```
WIN 1 Mailbox 1                          256.304k
WIN1_exec
QSpread                 |  62812|30 Jun 99 11:50|
MenuConf_INF            |   1936|26 Jun 99 15:01|
FiFi                    |  25268|06 Mai 99 19:12|
QMAKE                   |  27666|10 Nov 98 14:36|
SYSMON_TT               |   2722|15 Jun 98 17:47|
sysmon                  |   2722|18 Mai 99 11:16|
xchange                 | 187768|17 Mai 98 22:19|
qascade                 |  16466|15 Mai 98 19:45|
qascade_rc              |    587|15 Mai 98 19:45|
netmon                  |   1190|02 Mai 99 17:47|
Menuconfig              |  23770|11 Nov 98 19:29|
QLINK                   |  22560|26 Apr 98 23:35|
QD                      |  56608|25 Mär 98 19:44|
QMAC                    |  34452|22 Mär 98 12:03|
WinEd                   |  17260|13 Feb 98 20:47|
qeyes                   |   9240|12 Dez 97 20:11|
qfv                     |  91916|27 Okt 97 21:22|
WINSel                  |  12980|19 Jun 99 10:20|
xchange_mono            | 192006|18 Okt 97 18:00|
FI2config_obj           |  83950|06 Jun 99 00:10|
DISKCODE_EXE            |   9648|02 Okt 97 15:56|
```

```
RAM 8 RAM8                               2.178k
RAM8_
___SET_ALTER_LISTING___              |
SET_ASM                              | 16
___EXCEPTIONS_ASSEMBLER_LISTING___   |
EXCEPTN1_ASM                         |  5
___READ_THIS_FILE_PLEASE___          |
README_doc                           |  6
___CUEDARK___                        |
cuedark                              |  8
MenuConfig                           | 26
mdl_paper_bas                        |
mdl_paper_exe                        |
mdl_STAR_bin                         |
mdl_helmut_bas                       |  8
mdl_paper_asm                        |
CueDark_doc                          | 12
CueDark_txt                          |  9
___MORE_CUEDARK_MODULES___           |
ANA_CLOCK_bas                        |  5
ANA_CLOCK_obj                        | 11
ANA_CLOCK_cfg                        |
BUBBLES_bas                          |  3
BUBBLES_obj                          |  9
BUGS8_obj                            | 17
BUGS4_obj                            | 17
CIRCLES_bas                          |  1
CIRCLES_obj                          |  1
DEBRIS_bas                           |  1
DEBRIS_obj                           |  7
DIGITAL_obj                          |  6
DIGITAL_bas                          |  2
FISHTANK_obj                         | 11
```

QLTdis - part three

Norman Dunbar

In the last instalment of QLTdis, we didn't get much in the way of coding done. This installment makes up for that. There have been a few bits added here and there to the original code files that you either typed in from part one or loaded in from the cover disc. From here on, there is no cover disc option so get your typing fingers ready.

The 'working' code we have so far is not very useful. All it does is ask for a start address, end address and a printer device, and that is about it. This issue, we add a few utilities to the code, tidy up what we already have working and make a start on the central part of the dissassembly routines using the information in the previous article. So here goes with a summary of the code changes so far.

GWASL BUG!

Shock horror. I have found a bug in George Gwilt's assembler which we are using for this series. At the moment this will not affect us, but here are the details, just in case.

When assembling a TRAP #15 instruction, the code generated is for TRAP #0 and an error message is produced which says 'ERROR Wrong SIZE'. This causes the generated code to lock up as it switches into supervisor mode when TRAP #0 is executed.

As a workaround, if you want to put a TRAP #15 in your code, put DC.W $4E4F instead.

EQUATES_ASM

Added equate for oops. This was a simple change, simply add the following line to the file:

```
oops   equ -1   ; General error code for sub-routines
```

In my version, I added this under the line that defined 'linefeed'. In practice, it makes not a jot of difference where you put it. All we have done is set up a general purpose error value so that any of the routines that need to, can simply have a couple of lines such as this

```
        MOVEQ  #oops,DO     RTS
```

to set up an error exit from the routine.

INIT_ASM

Just above the line with the label 'get_start' there is a line that reads:

```
        bsr  cls
```

Please comment this line out by inserting an '*' in the first character of the line so that it reads as follows:

```
*    bsr  cls
```

This makes the work we are going to do later easier to see.

Further down, change the line that currently reads :

```
got_start nop       ; WE COME BACK TO THIS NEXT TIME
```

to the following 3 lines:

```
got_start
        bsr      get_addr         ; Convert user input to
                                  ; an address
        bne.s    get_start        ; Illegal entry
        move.l   a2,pc_addr(a4)   ; Store start address
```

Then change the line that currently reads :

```
got_end  nop       ; WE COME BACK TO THIS NEXT TIME
```

to the following 3 lines:

```
got_end
        bsr      get_addr        ; Convert to an address
        bne.s    get_end         ; Illegal entry
        move.l   a2,pc_end(a4)   ; Store end address
```

The NOP instructions I left as a placeholder should be removed from both of these code fragments.

Just above the label 'MOVE_PRNAME' there is the following code:

```
        lea      pr_dev,a0       ; Storage for printer
name
        move.w   d1,(a0)+        ; Save filename length
        subq.w   #1,d1           ; Adjust for dbra loop
*--------------------------------------------------------
* At this point:
*
* A1.L = first character in input buffer for printer name
* A0.L = first character position in pr_dev buffer
* D1.W = Number of character in filename minus 1
*--------------------------------------------------------

move_prname
        move.b   (A1)+,(A0)+   ; Move a single character
        dbra     d1,move_prname ; And the rest
```

All of the above lines should now be replaced by the following 3 lines:

```
move_prname
        lea      buffer,a1   ; Where printer name is now
        lea      pr_dev,a2   ; Destination for printer
                             ; name
        bsr      str_copy    ; Copy printer name
```

This is much simpler and uses a new utility that we will be writing later on in this article. The utility is str_copy and will be appearing in utils_asm. (See below)

Further down in this file, there are another two placeholder NOP instructions. These have also been replaced as follows:

The first is located above the label 'SHOW_END' and should be replaced with the following 3 lines:

```
        move.l   pc_addr(a4),d4 ; Get the start address
        bsr      print_hex      ; Convert & print start
                                ; address
        bsr      line_feed      ; Print a new line
```

JOCHEN MERZ SOFTWARE

Im stillen Winkel 12 D-47169 Duisburg
Tel. 0203 502011 Fax 0203 502012
http://www.j-m-s.com/smsq/index.htm

Ink Attack

High-quality alternative ink cartridges for EPSON

Stylus Color 600, 800, 1520	Color	DM 21,90
Stylus Color 800, 1520	Black	DM 15,90
Stylus Color 500	Color	DM 19,90
Stylus Color 400, 500, 600, Photo	Black	DM 15,90
Stylus Color 300	4-Color	DM 22,90
Stylus Color Photo, 700, EX	5-Color	DM 27,90
Stylus Color 440, 640	Black	DM 17,90
Stylus Color 740	Black	DM 17,90
Stylus Color 440, 640, 740	Color	DM 23,90
Stylus 800 (old model)	Black	DM 12,90
Stylus Color II, IIs, 825, 1500	Color	DM 18,90
Stylus Color II, IIs, 825, 200	Black	DM 13,90

High-quality alternative ink cartridges for CANON

BJC-70, BJC-80	Black	DM 8,99
BJC-70, BJC-80	Color	DM 10,99
BJ 300, 330	Black	DM 11,99
BJC 600, 610, 620		
Black, Cyan, Magenta, Yellow	EACH	DM 7,99
BJC 800, 820, 880		
Black, Cyan, Magenta, Yellow	EACH	DM 11,99
BJC 4000, 4100, ...	Black	DM 8,99
BJC 4000, 4100, ...	Color	DM 12,99
BJC 7000 Series	Color	DM 32,99
BJC 7000 Series	Photo	DM 36,99
BJC 7000 Series	Black	DM 34,99

Other printing stuff

CD-Labels (10 pages with 3 labels) white	DM 6,99
Inkjet paper for DOUBLE-SIDED printing A4, 360 dpi, 80g, 200 sheets	DM 12,99
Inkjet paper for DOUBLE-SIDED printing A4, 720 dpi, 250g, 50 sheets	DM 22,99
Inkjet paper single-sided printing A4, 1440 dpi, 105g, 50 sheets	DM 17,99
Inkjet glossy paper A4, 1440 dpi, 10 sheets	DM 14,99
same as above, but 50 sheets	DM 49,99
Inkjet photo paper A4, 1440 dpi, 160g, 10 sheets	DM 18,99
Inkjet glossy paper for DOUBLE-SIDED printing A4, 360 dpi, 135g, 100 sheets	DM 18,99
same as above, but 100g, 100 sheets	DM 18,99

Minimum order

3 Items or DM 60,-

TERMS OF PAYMENT

Postage and package **[Germany]** DM 8,99 (if total value of goods is up to DM 50,- then only DM 5,99). **[Europe]** DM 14,50 (if total value of goods is up to DM 50,- then only DM 9,50). **[Overseas]** between DM 14,50 (1 item) and DM 35,- (maximum). All prices incl. 15% V.A.T. (can be deducted for orders from non-EEC-countries). E&OE. Cheques in DM, £'s, Eurocheques and Credit Cards accepted.

EUROCARD. VISA MasterCard

And then again, above label 'SHOW_PRNAME' change the NOP into the following 3 lines:

```
        move.l   pc_end(a4),d4  ; Get the end address
        bsr      print_hex      ; Convert & print end
                                ; address
        bsr      line_feed      ; Print a new line
```

This is the end of the INIT_ASM changes. The code changes made in this file make calls to other routines we don't yet have in our various files. These are typed into UTILS_ASM.

UTILS_ASM

First an error - did anyone spot it ? There is a line in utils_asm which defines our input buffer. It reads as follows:

```
buffer   ds.w    16   ; 60 chars for input plus 1
                      ; word for size.
```

Well, as far as I remember, 16 times 2 is only 32 so we need to change this to the following:

```
buffer   ds.w    31   ; 60 chars for input plus 1
                      ; word for size.
```

Next, I have added the routine which allows you to actually type in an address for the start and end addresses when you run the program. In its present state, it just accepts anything you type. Add the following code to the end of UTILS_ASM:

```
* ===================================================
* Get_addr:
* ===================================================
* 1. Convert the users input from text to long word.
*
* Returns Z set if no errors, unset if errors.
* Returns A2 = Address converted.
* ===================================================

get_addr
        clr.l    d7          ; Converted address goes here
        clr.l    d1          ; Need long sized for
                             ; additions later
        move.w   -2(a1),d0   ; Counter of chars in buffer
        cmpi.b   #'$',(a1)   ; Hex address?
        beq.s    get_hex_ad  ; Yes - convert from hex
        bra.s    gd_more     ; Thanks George !
```

This routine is the controlling part. Any time the program requires that you type in a valid hexadecimal or decimal address, this routine will be called to validate your entry. Entering an address can be done in either hex or decimal. Tradition dictates that all hex numbers are prefixed by a dollar sign ($) to distinguish them from decimal addresses. So when we are using QLTdis, if we want to disassemble from address 1024 (dec) we simply enter 1024 when prompted.
Should we wish to enter its hex equivalent which is $400, we must prefix this with a '$' to show that

QBOX-USA BBS

Operating since 1993 on a Sinclair QL from Utica, Michigan, USA
Supporting *ALL* Sinclair and Timex users
Message and File Areas for QL, Z88, Spectrum, TS2068, ZX81, TS1000
Modem speeds 300 bps to 33.6k bps supported
24 hour operation - call anytime
810-254-9878

it is a hex address and not decimal 400. We would therefore type $400 when prompted.
The above routine first clears out its working registers and obtains a count of the number of characters in the input buffer into D0.W. Normally this routine is called directly after a call to the 'input' routine and so A1.L is pointing to the very first character in the buffer and not at the word which defines the length of the text in the buffer. We get the length from the address which is A1 -2.
If the first character in the buffer is a dollar, then we skip off to the routine 'GET_HEX_AD' and read the remaining characters as hex data, otherwise we skip to the DBRA instruction at label 'GD_MORE'. This simply adjusts the value in D0.W and terminates the routine if it is now -1.
The following code deals with obtaining a decimal address from the user's input. This code follows on from the above.

```
* ====================================================
* Get_dec_ad :
* ====================================================
* 1. Convert the users input from decimal text to long word.
*
* Expects A1 to point to first character in the input buffer.
* Expects D7.L to be zero.
* Expects D0.W to be the count of characters in the buffer.
*
* Returns Z set if no errors, unset if errors.
* Returns A2 = Address converted.
* ====================================================
get_dec_ad
        move.b  (a1)+,d1  ; Fetch a character
        cmpi.b  #'0',d1   ; Digits only
        bcs.s   not_good  ; C set means D1 < '0' =
                          ; not valid
        cmpi.b  #'9',d1   ; Digits only
        bhi.s   not_good  ; (unsigned) D1 > '9' =
                          ; not valid
        subi.b  #'0',d1   ; From '0' - '9' to 0 - 9
        lsl.l   #1,d7     ; Multiply current total by 2
        move.l  d7,d2     ; Save it for later
        lsl.l   #2,d7     ; Now multiply by 4 (= D7 * 8)
        add.l   d2,d7     ; D7 now = D7 * 10
        add.l   d1,d7     ; And add the latest digit
gd_more
        dbra    d0,get_dec_ad ; And the rest
        move.l  d7,a2         ; Return address in A2.L
        clr.l   d0            ; No errors
        rts

not_good
        moveq   #oops,d0      ; Bad number flagged
        rts
```

The routine simply gets each character from the buffer, and compares it with a digit zero (not the value zero !) to see if it might possibly be a digit. If the Carry flag is set, then the character was below '0' in the ASCII character set. This is invalid so we branch to 'NOT_GOOD' where we process the error and exit without an address in A2. We must test the Z flag on return from GET_ADDR to make sure that the value in A2 is valid.
Next we must check if the character is less than or equal to a digit '9' (again, not the value 9) and if greater, we again have an invalid character so it is rejected accordingly.
We must now have a valid character in the range '0' to '9'. In order to convert this into a proper value, we simply subtract the value of '0' from the character. So, we have a value in D1.L which we need to add in to our running total in D7.L - how does this work?
When the user typed '1024' this was stored in the buffer. We are now processing the characters in the buffer from the beginning to the end (or until an error). This means that we start with the '1', then the '0' then the '2' and finally the '4'. We must exit from the routine with A2.L holding the value 1024. How?

As we convert each digit into a decimal value from 0 to 9, we simply multiply the running total (A0) by 10 and add the new digit to the total. The processing proceeds as per the following table:

Character in buffer	D7.L before	D7.L after
'1'	0	0 * 10 + 1 = 1
'0'	1	1 * 10 + 0 = 10
'2'	10	10 * 10 + 2 = 102
'4'	102	102 * 10 + 4 = 1024

The remainder of the code does the multiply and add. Using unsigned values (all addresses are from zero upwards!) we simply shift D7.L to the left by one place. This is the same as multiplying by 2. This value is stored in D2 as we will need it later. D7.L is then multiplied by 4 by shifting left another 2 places. We now have D7 = old D7 * 8 and D2 = old D7 * 2, adding these together gives us the value of old D7 * 10. D1 is then added to give the new running total.
At the end of the buffer, D7 is copied to A2 and D0 is cleared if no errors occurred and we return to when we came. If there were any errors - invalid digits in the bufffer - the code at 'NOT_GOOD' simply moves -1 into D0 which clears the Zero flag to signal an error.

Getting addresses in decimal is actually more difficult than getting them in hex. The following routine is called whan we find a '$' as the first character in the buffer.

```
* ====================================================
* Get_hex_ad :
```

```
* ===================================================
* 1. Convert the users input from Hexadecimal text to long word.
*
* Expects A1 to point to first char in the input
* buffer which is the '$'
* before the first hex digit.
* Expects D7.L to be zero.
* Expects D0.W to be the count of characters in the buffer.
*
* Returns Z set if no errors, unset if errors.
* Returns A2 = Address converted.
* ===================================================
get_hex_ad
        addq.l  #1,a1       ; Skip the '$' in the
buffer  subq.w  #1,d0       ; Reduce counter for '$'
        bra.s   gh_more     ; Thanks George !

get_hex move.b  (a1)+,d1    ; Fetch a character
        cmpi.b  #'0',d1     ; Do we have a digit ?
        bcs.s   not_good    ; Less than '0' is invalid
        cmpi.b  #'9',d1     ; Digits only
        bls.s   digit_ok    ; Got a digit if <= '9'

        bclr    #5,d1       ; UPPERCASE (possible)
letter  cmpi.b  #'A',d1     ; Letters A - F only
        bcs.s   not_good    ; Less than 'A' is invalid
        cmpi.b  #'F',d1
        bhi.s   not_good    ; Greater than 'F' is
invalid

hex_ok  subq.b  #7,d1       ; Adjustment for letters

digit_ok
        subi.b  #'0',d1     ; From '0' - '9' to 0 - 9
        lsl.l   #4,d7       ; Multiply current total by 16
        add.l   d1,d7       ; And add the latest hex digit

gh_more dbra    d0,get_hex  ; And the rest
        move.l  d7,a2       ; Return the address in A2.L
        clr.l   d0          ; No errors
        rts
```

This routine has certain similarities to the above one. There is more initial checking carried out as we need to have a digit or a letter 'A' to 'F' or 'a' to 'f' in order to have a valid hex address. Additionally, the buffer pointer in A1 is still pointing at the dollar and D0.W includes the dollar sign in its count of characters in the buffer.
First we increment A1.L so that we skip the '$'. D0 is reduced by 1 to account for this character being removed. We then skip over the code first time to avoid any of the famous DBRA problems if D0 is now zero (ie the user simply typed '$' and ENTER when prompted.)
The first part of the testing is exactly the same as above, but if the character is not a digit, we must then check for a letter. In order to reduce the comparisons we need to do, bit 5 of the character is set to 0 by the BCLR #5,D1 instruction to convert a possible letter from lower case to upper case.
In case you were wondering, the letter 'a' has ASCII value of 97 and 'A' has the value 65. The difference is 32. In Binary these values are:

```
'a' = 97 = $61 = 0110 0001
'A' = 65 = $41 = 0100 0001
bit number  = 7654 3210
```

By resetting bit 5 (counting from the right end with bit 0 being first) we have converted 'a' into 'A' and if it was already 'A' then it has not changed.
We now check that we have a character in range 'A' to 'F' and if not, exit via the 'NOT_GOOD' routine above.
There are 7 characters after the digit '9' and before the letter 'A' in the ASCII character set. To adjust our letter characters we subtract 7 from the ASCII code so that 'A' which was 65 is now 58 and 'F' becomes 63.
At this point, we subtract the ASCII for '0' from our character to convert it to a value between 0 and 15, the ASCII for '0' is 48, so '0' become 0 and '9' which is 57 becomes 9. 'A' which was adjusted to 58 becomes 10 and 'F' becomes 15. Easy or what?
Finally, in a manner similar to decimal addresses, the running total is multiplied by 16 and the new character value added in. Multiplying by 16 is simply a case of shifting left by 4 places. Again, we exit with D0 set to zero (and the Z flag set) and A2.L holding the address required.

Type the following routines into UTILS_ASM. These are routines that convert the binary addresses and numbers in a register back into human readable format.

```
* ===================================================
* hex_l:
* ===================================================
* Convert a 4 byte value in D4.L to Hex in a buffer.
* Use the input buffer for the output and DOES NOT
* store the length word !
*
* Expects D4.L to hold the value.
* ===================================================
hex_l   swap    d4          ; $ABCD -> $CDAB in D4
        bsr.s   hex_w       ; Convert the $AB part first
        swap    d4          ; $CDAB -> $ABCD again
*       drop into hex_w to convert the $CD part

* ===================================================
* hex_w:
* ===================================================
* Convert a 2 byte value in D4.W to Hex in a buffer.
*
* Expects D4.W to hold the value.
* Expects A1.L to point at the buffer.
* ===================================================
hex_w   ror.w   #8,d4       ; $DE -> $ED in D4
        bsr.s   hex_b       ; Convert the $D part first
        rol.w   #8,d4       ; $ED -> $DE again
*       drop into hex_b to convert the $E part

* ===================================================
* hex_b:
* ===================================================
* Convert a 1 byte value in D4.B to Hex in a buffer.
```

```
*
* Expects D4.B to hold the value.
* Expects A1.L to point at the buffer.
* ===================================================
hex_b   ror.b   #4,d4       ; Swap lower and higer nibbles
        bsr.s   hex_nibble  ; Print high nibble first
        rol.b   #4,d4       ; Swap back again
*       drop into hex_nibble to print the lower nibble

* ===================================================
* hex_nibble:
* ===================================================
* Convert a 4 bit value in D4.B to Hex in a buffer.
*
* Expects D4.B to hold the value.
* Expects A1.L to point at the buffer.
* ===================================================
hex_nibble
        move.b  d4,-(a7)    ; Save value in both nibbles
        andi.b  #$0f,d4     ; D4.B now = 0 to 15
        addi.b  #'0',d4     ; Now = '0' to '?' (ascii only)
        cmpi.b  #'9',d4     ; Is this a digit ?
        bls.s   nib_digit   ; Yes
        addi.b  #7,d4       ; Add offset to UPPERCASE letters

nib_digit
        move.b  d4,(a1)+    ; Store in buffer
        move.b  (a7)+,d4    ; Restore original value
        rts
```

Starting at the easy part first at HEX_NIBBLE. If D4.B holds the value $F7 then we convert this from binary in the register to character data in a buffer somewhere. We do this as follows:
Preserve the current value in D4 on the stack. We need it again later unchanged. Next we mask off the highes 4 bits in D4.B leaving only the lowest 4 bits. Using our original value, D4.B has just become $07 from $F7.
We convert this to a character by adding the ASCII code for '0' to it. This changes D4.B from $07 to $37 which just happens to be the character code for '7'. If this is in range '0' to '9', we are ok, and simply store it in the buffer pointed to by A1.L and retrieve the original value of D4 from the stack.
Our original value in D4.B was $F7, and we have just put the '7' part into the buffer.
Working backwards, to HEX_B, we have a routine to store both characters in D4.B in the buffer. As you have just seen, HEX_NIBBLE always stores the lowest nibble in the buffer. We as humans like to see the value '$F7' written as F7 and not as 7F. This routine takes the current value (of $F7) and rotates it 4 bits so that it becomes $7F instead.
We then call HEX_NIBBLE to deposit the lowest nibble - which is now the 'F' into the buffer. On return from HEX_NIBBLE, we rotate D4.B again so that it once more becomes $F7 and then drop directly into HEX_NIBBLE to deposit the lowest nibble - which is now the '7' into the buffer.
This is exactly what we want, the value in D4.B is $7F and the buffer now has the characters '7' and 'F' in it.

Moving further backwards, HEX_W converts a word sized value in D4 into the buffer. Assume D4.W is holding $1234 at this point. First we rotate D4.W 8 bits to make the value $3412, then we call the HEX_B subroutine to deposit the '1' and '2' into the buffer in that order. On return, we rotate D4.W back again so that it once more becomes $1234. Following this, we fall into HEX_B and this causes the characters '3' and '4' to be stored in the buffer. Once more, we have exactly what we want in the buffer - '1', '2', '3' and '4' in that order.
And finally, HEX_L converts the value in D4.L in exaclty the same manner. First the two words of D4 and swapped over, HEX_W is called to deal with the lowest word (which was the highest), and then the words are swapped back again prior to falling through into HEX_2 again to do the original lowest word.
Think about it for a while and it becomes clear. Hopefully I described it correctly and clearly so you shouldn't have too many problems with it. Try running through exactly what happens when you call HEX_B, HEX_W or HEX_L, writing down the values in all the registers being used and what is in the buffer etc until you have it.
The next routine is called PRINT_HEX and is used to convert the value in D4.L from binary to hex in a buffer, then print it to the screen. This is used when we show the user of QLTdis the chosen start and stop addresses. We use the user input buffer as a temporary workspace for the conversion. D4.L is converted into 8 hex digits and then printed by the 'PROMPT' routine we have used since part 1 to print a string to the screen. This routine will also be used to print the address of the instruction being disassembled when running the program.

```
* ===================================================
* print_hex:
* ===================================================
* Convert D4 into 8 hex characters, then print it to
* the channel in A0.L
*
* Expects D4.L to hold the value.
* Expects A0.L to hold the channel id.
* ===================================================
print_hex
        lea     buffer,a1   ; Output buffer for address
        move.w  #8,(a1)+    ; We know the result is 8 bytes
        bsr     hex_l       ; Convert 4 bytes to text
        lea     buffer,a1   ; Text to print
        bsr     prompt      ; Print it
        rts                 ; All done. (Error code in D0)
```

The next issue will continue where this issue left off due to the large size.
Norman Dunbar would like it to be known that his wife Alison would like to be mentioned in QL Today and credited with giving him 5 hours time off (for abnormally good behaviour) in order to write this article!"

System Variables Hint

David Denham

A friend using an old BASIC program of mine commented that it wouldn't work on his Minerva-equipped QL, because I'd included a reference to a system variable, which caused the program to fail when he ran his computer in second screen mode, as this causes the system variables to move to a different location in memory. This would also apply to an Aurora I suppose, when using the higher resolution modes with SMSQE.
You can use basic extensions like Dilwyn Jones' Display_cde to return the address of the system variables, but this has three drawbacks:

1. It relies on the user of the program having those extensions ***[they are public domain files, anyone can use them or give them away with a program and they work on all systems - Editor]***
2. Even if such extensions can be included with a program given away or sold, there is the overhead of having to write a boot file to load the extensions, etc, unless you QLiberator compile the program and build in the machine code extension.
3. Some extensions for this purpose, such as those built into SMSQE only work on that platform, if the program is run on an old QDOS system there is no guarantee it would work.

The function VER$ can be used to identify a Minerva or SBASIC system and a variant of VER$ on these systems can be used to tell you the system variables address. If running on neither of these systems, the chances are it's (a) a very rare system, or (b) an original QDOS QL where the system variables are at a 'known' address. Using this information I came up with the following function in BASIC which will cope with the majority of situations where you need to know this address.

```
1000 DEFine Function SYSTEM_VARIABLES
1010 LOCAL v$
1020 v$ = VER$
1030 IF v$="JSL1" OR v$="HBA" THEN
1040 RETURN VER$(-2) : REMark system variables
     base
1050 ELSE
1060 RETURN 163840 : REMark original QL
     address
1070 END IF
1080 END DEFine
```

Using the IF clause helps to prevent errors caused by the fact that original QL ROMs won't recognise VER$(-2).
■

Printer Control Codes - A Dreaded Subject? - Part 4

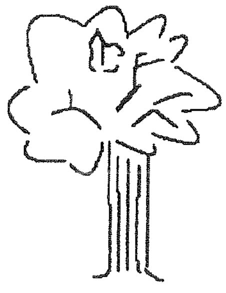
Dilwyn Jones

I had intended to finish this series in the last issue. But some readers asked me to put together a sample Quill printer driver for HP Deskjet printers, as that seems to be the printer control code set which gives users greatest difficulty.
I have been using this driver with a HP Deskjet 320 portable printer for a while now and it seems to work quite well. You may have to make slight changes for different Deskjet models if you wish to take advantage of additional features not available to me.
I have tried to stick to 'standard' HP control codes so this stands a good chance of working on most HP inkjet printers.
Elsewhere in this issue, Jochen has asked me to document the Psion PRINTER_DAT format for Quill, so that you have the information to permit your programs to make use of PRINTER_DAT as well.
Here is the necessary information to enter into INSTALL_BAS to create a Quill printer driver for Hewlett Packard Deskjet printers. It assumes the printer is used on SER1. As I was unable to get both subscript and superscript to work (both came out the same on my Deskjet 320) I decided I'd make use of superscript to print italics characters instead. This is shown as superscript on screen in QL normally, so it stands out, but if you have software to patch Quill fonts (e.g. a program by Howard Clase in the Quanta library) you might like to rise to the challenge and create an italics font for Quill superscript!
Some of the translates especially assume an American or UK character set, and switch to other character sets on the printer to get international characters. Beware of this in case it varies from country to country - I have no way of knowing. Perhaps users in Europe could let us know if this driver does still work on European printers.

```
PRINTER PORT          : SER1
PARITY                : NONE
BAUDRATE              : 9600
LINES PER PAGE        : 64
CHARACTERS PER LINE   : 80
FORMS TYPE            : CONTINUOUS
END OF LINE CODE      : 10
PREAMBLE CODES        : 13,27,40,56,85,27,38,107,50,71
POSTAMBLE CODES       : 12
BOLD ON CODES         : 27,40,115,51,66
BOLD OFF CODES        : 27,40,115,48,66
UNDERLINE ON CODES    : 27,38,100,49,68
UNDERLINE OFF CODES   : 27,38,100,64
```

RWAP
SOFTWARE

2000



Make History

Create Your Family Tree to Cover
the Last and the Next Millenium

QL Genealogist v3.24 £20

Genealogy for Windows £50

Special Offer

QL Payroll v3.5 £5
QL Cash Trader v3.5 £5
Both £8

QHelp v1.03 £10
SideWriter v1.08 £10
Q-Route v1.08C £25

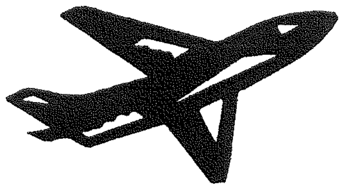
Adventure Pack

Return to Eden	The Prawn	Lost Kingdom of Zkul	Nemesis MKII	Horrorday	West

All Six Adventures **Only £35**

Frightened of the 21st Century?

Fly Away From
It All..



FlightDeck v1.04

£10

RWAP Software, 4 Anvil Crescent,
Coseley, West Midlands
WV14 8GA
TEL: 01902 836888
E-MAIL: rwap@qbranch.demon.co.uk

Cheques in £sterling
payable to 'R.Mellor'

```
SUBSCRIPT ON CODES      : 27,40,115,54,86
SUBSCRIPT OFF CODES     : 27,40,115,49,50,86
SUPERSCRIPT ON CODES    : 27,40,115,49,83
SUPERSCRIPT OFF CODES   : 27,40,115,48,83
TRANSLATE 1 CODES       : 96,175
TRANSLATE 2 CODES       : 136,181
TRANSLATE 3 CODES       : 168,180
TRANSLATE 4 CODES       : 132,206
TRANSLATE 5 CODES       : 164,218
TRANSLATE 6 CODES       : 145,27,40,115,53,72
TRANSLATE 7 CODES       : 146,27,40,115,49,48,72
TRANSLATE 8 CODES       : 127,99,8,79
TRANSLATE 9 CODES       : 163,27
TRANSLATE 10 CODES      : 188,8
```

Some Notes on the above Driver Codes

END OF LINE CODE

Normally, Deskjets require a carriage return and linefeed (codes 10 and 13), which Quill would be able to provide, or the printers can be set up to generate the CR (13) automatically. I have chosen to manually place the printer in auto-CR mode so that I can use it from a PC as well without having to reset the internal presets each time. The QL normally sends a linefeed only, so I have set up PRINTER_DAT to do the same, and the preamble codes set the printer to generate the CHR$(13) carriage return itself on receipt of a linefeed CHR$(10)

PREAMBLE

The first 13 is simply a carriage return sent to ensure that the print head always starts printing at the left margin. Some software has the annoying habit of leaving the print head halfway across the line, so the first line of a new page of text is printed starting halfway across the page.
`27,40,56,85` places the printer in HP Roman 8 character set, the one best suited to my needs.
`27,38,107,50,71` is a set of codes which make the printer generate a carriage return in addition to the linefeed sent by the QL. In fact, this can be a useful set of codes to put on a hotkey or an altkey:

```
ERT HOT_CMD('5','OPEN #3,SER1:BPUT #3,27,
38,107,50,71:CLOSE #3')
```
or
```
ALTKEY '5','OPEN#3,SER1:BPUT #3,27,38,107
,50,71:CLOSE #3'
```

If you use the HOT_CMD function, it sets up a hotkey which picks BASIC and sends a command to send those codes to the printer. The altkey definition is the same, except that it does not pick basic - the text is simply sent to the current job.

POSTAMBLE

This simply sends a formfeed or CHR$(12) to the printer to ensure that the last page printed is ejected from the printer - my Deskjet 320 has the optional sheet feeder and there is nohing worse than having to try to free a last page of text which gets stuck in the printer once the paper has crinkled because I forgot to remove it, or simply left it for hours to finish printing long documents!

BOLD ON AND OFF

This sets standard bold printing. If you prefer extra bold, use `27,40,115,55,66` which only works on some printers as it is an option. The same BOLD OFF sequence is used for either.

UNDERLINE ON AND OFF

This uses standard single fixed underline. Deskjets support 4 types of underline, which you can experiment with by using 50 or 51 or 52 in place of the 49 in the underline on sequence. The same code is used to cancel all the underline variations.

SUBSCRIPT ON

Although the programmer's manual I have for the Deskjet 500 lists code sequences for subscript or superscript placement of text, I have been unable to get this to work on my Deskjet 320 at least.
So I cheated and used the code set for 6 point high text as a sort of subscript text instead.
Thankfully, Quill has separate cancel sequences for subscript and superscript, so I can use superscript for something else.

SUPERSCRIPT

As I couldn't get this to work on my printer, I used this for italic sloping text instead.

TRANSLATES

A common problem is trying to get the British/Irish pound symbol to print properly. This can be a frustrating experience, as it may only exist in a U.K. character set, or be at a different position in the other national character sets. On a British QL, a pound symbol is CHR$(96). On most printers, sending CHR$(96) results in a backtick or accent character, clearly not what we want (well, perhaps we could join the Euro instead of awkward pound symbols, but that's another story - Dietrich Buder explained recently how to print Euros)
On some printers, the pound symbol in the UK character set replaces the hash symbol at CHR$(35), but on the character set I use on my

Deskjet, there is a pound symbol at CHR$(175), so all I have to do is translate code 96 to code 175.
If the character set you choose to use does not have a pound symbol, you may have to add codes here to temporarily switch to a character set that contains a pound symbol, send the code for pound in that set, then send the codes to switch back to the original character set - a useful hint sometimes.

As I occasionally correspond with readers in French or German speaking countries, I need to be able to print some accented characters, vowels especially. This also comes in useful for Welsh correspondence.
The first such accented character to be handled is a sedilla c (ç) - CTRL SHIFT 9 on the laptop on which I'm writing this in QPC. This is CHR$(136) on the QL but CHR$(181) on the printer.
Similarly for upper case C (Ç) CHR$(168) on the QL. It is sent as 180 to the printer.

Translates 6 and 7 are a bodge. Since Quill does not allow double width printing, I use the keystrokes CTRL 1 and CTRL 2 to switch the printer into 5 characters per inch mode. These appear as ê (double width 1) and ï (double width off), which restricts my use fo these characters. Obviously, you can use any pair of characters you don't normally use - I chose CTRL 1 because it was easy to remember and because the ê character implies 'e for enlarged'

Translate 8 is another bodge to try to handle the copyright symbol © which is another QL character which can be difficult to print. Some users give up and translate it to a fairly similar looking @symbol on some printers. My approach is to translate CHR$(127) into a c followed by a backspace to move the printhead back and overprint an upper case O, the closest I can get on my printer. You could if you really wanted to translate ©into the 3 separate characters (C and), although this can have funny effects on line lengths, because 80 character lines on screen could be 82 characters wide by the time they get printed, causing line overspill to the next line. Though as the symbol is often used in short lines such as

Copyright ©1999 Dilwyn Jones

this may not be an issue in some cases, though it does serve to illustrate how you sometimes have to think laterally to find ways of getting characters to translate satisfactorily.
Translate 9 converts CTRL SHIFT C - the É character - into an ESC character for the printer. This is useful in two ways - if you need to find a way of sneaking certain characters or functions into a document, a rare character or switching print pitch or whatever, it can be useful to directly insert characters, and have a character to correspond to ESC on screen, and É to me looks like a symbol which implies ESC! So if I wanted to double underline something, I could insert the following lines within the text, either side of the text to be underlined (again beware of wrong justification or line splits because of extra unprinted characters)

É&d2DDOUBLE UNDERLIND HEADINGÉ&d@

This can be pretty tricky stuff to work out - ignore it if you are uncertain of how it works.

Finally, in order to generate accented characters I sometimes find it useful to be able to overprint a symbol over a letter, which means I need a symbol to represent backspacing. I chose the left arrow symbol CHR$(188) to do this - a left pointing arrow is shown in Quill to imply backspacing, so I can print a vowel followed by a backspace followed by a circumflex o←^ for example.

Things - Part 3

Jochen Merz

This time we will have a closer look at the things in your system, how they are organised in memory and how they are linked.
You will find a fairly short SBASIC program on the next page (Listing 1) which you can type in and which will list a few more details about the Things in your system. When you run it, it will display a list of all Things which are currently available. You can see in table 2 which Things can be found in my system.
I should mention first, that hacking through a list like the Thing list (or other lists, like the Device Driver list etc.) should be done in Assembler while the program is running in Supervisor mode. This ensures, that other jobs cannot link or unlink Things while we scan the list. We cannot do this in SBASIC, but we as long as you don't add and remove Things while the program runs

nothing will happen. And I doubt you are permanently adding and removing Things while looking at them anyway.

Things are stored in a linked list. New Things are added at the top of the list. A system variable
`sys_thgl equ $b8`
points to the first entry in the list. The strange PEEK_L with the two exclamation marks reads a longword from the system variables at offset hexadecimal B8 from the start of the system variables.
In "ordinary" QDOS, you could write
`PEEK_L(HEX("280B8"))`
instead.
The address read from this variable is a pointer to the first Thing in the Thing-list. Every Thing points to the next one, so it is very easy to scan the list top to bottom. The list ends when the next pointer does not point to anything anymore but contains the value 0 instead. If you have a look at table 1, then you will see an explanation of the structure to which this pointer points.
The next three addresses will be filled in by the routine which links the Thing into the list, so you do not need to worry about them when you create a Thing - the Usage List is a linked list of all jobs using this Thing at a given time, and the other two routines should not be touched at all.The pointer to the Thing itself needs to be filled in - nobody but you know where the Thing is actually located.
The next four routines should be provided by you so that

Table 1

```
; Thing linkage block

th_nxtth equ $00 ; long   link to NeXT THing
th_usage equ $04 ; long   thing's USAGE list
th_frfre equ $08 ; long   address of "close" routine for FoRced FREe
th_frzap equ $0c ; long   address of "close" routine for FoRced ZAP
th_thing equ $10 ; long   pointer to THING itself
th_use   equ $14 ; long   code to USE a thing
th_free  equ $18 ; long   code to FREE a thing
th_ffree equ $1c ; long   code to Force FREE a thing
th_remov equ $20 ; long   code to tidy before REMOVing thing
th_nshar equ $24 ; byte   Non-SHAReable Thing if top bit set
th_check equ $25 ; byte   CHECK byte --- set by LTHG
th_verid equ $26 ; long   version ID
th_name  equ $2a ; string name of thing
th.len   equ $2c ;        basic length of thing linkage
```

Listing 1

```
100 REMark List Things
110 :
120 listptr=PEEK_L(!!$B8)
130 chan=3:OPEN#chan,con
140 PRINT #chan;\"LINK     VERS S ADDRESS  TYPE NAME"
150 REPeat loop
160   IF listptr=0 THEN EXIT loop
170   cr_thgaddr=PEEK_L(listptr+$10)
180   PRINT #chan;HEX$(listptr,32)!
190   IF PEEK_L(listptr+$26)<>0
200    PRINT #chan;!PEEK$(listptr+$26,4)!
210   ELSE
215    PRINT #chan;"     ";
220   END IF
230   PRINT #chan;!"+-"((PEEK(listptr+$24)>$7F)+1)!
240   PRINT #chan;!HEX$(cr_thgaddr,32);' ';
250   cr_thgtyp=PEEK_L(cr_thgaddr+$4)
260   th_nam$=PEEK$(listptr+$2C,PEEK_W(listptr+$2A)):namlen=LEN(th_nam$)
280   IF namlen<20
290     th_nam$=th_nam$&FILL$(' ',20-namlen)
300   ELSE
310     IF namlen>20 THEN th_nam$=th_nam$( TO 20)
320   END IF
330   SELect ON cr_thgtyp
340    =0:PRINT #chan;'UTIL'!th_nam$!
350    =1:PRINT #chan;'EXEC'!th_nam$!
360    =2:PRINT #chan;'DATA'!th_nam$!
370    =$1000003:PRINT #chan;'EXTN'!th_nam$!
380    =$1000004:PRINT #chan;'EXTS'!th_nam$!
390    =-1:PRINT #chan;'VECT'!th_nam$\'  TH_ENTRY'!HEX$(PEEK_L(cr_thgaddr+$8),32)\'  TH_EXEC
     '!HEX$(PEEK_L(cr_thgaddr+$C),32)!
400    =REMAINDER :PRINT #chan;'    ';th_nam$!'UNKNOWN TYPE ('!HEX$(cr_thgtyp,32)!') '
410   END SELect
420   PRINT #chan
430   listptr=PEEK_L(listptr)
440 END REPeat loop
450 CLOSE#chan
```

Table 2

```
LINK     VERS S ADDRESS  TYPE NAME
000C3810      + 000C386E EXEC Calculator
00091E30 2.25 + 00091E62 EXEC Sernet
000917C0 1.30 + 00100BB4 EXEC Pic Viewer
00091760 3.31 + 00100B80 EXEC FileInfo II thread
00091700 3.31 + 0010091C EXTN FileInfo II extensio
000916A0 3.31 + 001008FE DATA FileInfo II history
00091640 FIv3 + 001008E0 DATA FileInfo II database
000915E0 ###@ + 001008BA UTIL FileInfo
00090FE0 1.00 - 000FC2D2 DATA DATAdesign mutex
00091890 3.14 + 000918CE EXTN DATAdesign.engine
0008B970 A.05 + 000E6D22 EXEC QD
0008B920 1.13 + 000DB5B6 EXTN Scrap Extensions
0008B8D0 7.57 + 000D8B1A EXTN Menus
00091320 1.03 + 00091360 UTIL Button Frame
000911F0 1.03 + 000CC6C2 EXTN Button Extensions
000911A0 1.04 + 000CF1BE EXEC Button_Sleep
00091150 1.02 + 000CF168 EXEC Button_Pick
00091100 1.01 + 000CF0B0 EXEC Hotjobs
000910B0 1.01 + 000CF098 EXEC Hotkeys
00091060 1.04 + 000CEE78 EXEC Channels
0008E7D0 1.02 + 000CEA44 EXEC Jobs
0008E780 1.25 + 000CCDB2 EXEC Files
0008E730 1.05 + 000CCAFC EXEC Sysdef
0008E6E0 1.02 + 000CCA26 EXEC Rjob
0008BBD0 1.02 + 000CC928 EXEC Pick
0008BB80 1.02 + 000CC7E6 EXEC Wake
0008BAA0 1.02 + 000CC7CE EXEC Exec
0008BA50 1.02 + 000CC2C6 EXEC Button
0008BA00 1.01 + 000CC068 EXEC Things
000580C0 2.15 + 000B8AD2 EXTN Jmon
0008A910 2.29 + 0008A948 UTIL HOTKEY
0008B4B0 2.05 + 007EAEAE EXTN DEV
0008A1A0 3.00 + 007EAC14 EXTN WIN Control
0008A050 3.08 + 0008A050 UTIL DV3
00089920 2.00 + 007F64E8 EXTN QVME
000898D0 2.09 + 0000AA6E EXTN Ser_Par_Prt
000581F0 2.11 + 0000A91E EXTN KBD
00059250 HBA  + 007F9E9E UTIL SBAS/QD
000591B0 HBA  + 007F9D5C EXEC SBASIC
00056C10 0.05 + 00056C42 VECT THING
  TH_ENTRY 00002FDA
  TH_EXEC  00002F40
```

can do whatever you like when a job tries to use, free or zap (force-free) this specific Thing. Also, a tidy-up routine can be provided (for example to unlink drivers or remove routines from an interrupt list etc.)
If the Thing can only be used from only one job at a time, then you should set the topmost bit of the non-shareable flag (marked with "-" in the listing output of the BASIC program).
Ignore the check byte, it is used for faster name comparison.
The version ID (or feature list) should be filled in by you, however.

The BASIC program reads the interesting bits from every Thing entry. In my example (table 2 again), you can see that the last Thing which has been added to the System is the Calculator (my HOTKEY definitions come last in my BOOT program!). Interesting to see, Calculator has no version number! QD, which is further down, has been LRESPRed earlier, but after QPAC2 (which links in a lot of Things ... everything from "Things" up to "Button Frame" if you read the table bottom-up.
Everything from the bottom up to "HOTKEY" is linked by the operating system SMSQ/E - this explains the order.

At the very bottom you find the special "THING" thing which defines the entry to the Thing system, but I explained this already. You will see how to use it in the future.

You can figure it out from the BASIC listing that, in my case, the Thing system variable points to $C3810, which points to $91E30, which points to $917C0 ... and so on until $56C10, which points to "0".
We also list the version number of each Thing. It is up to you what you fill in here - most people fill in four ASCII characters which define the Version. SBASIC, for example, has the version "number" HBA.
It is also possible to look at the 4 characters as 32 bits which specify "features" supported by the Thing. This means, you can't really look at "characters" but get funny patters or strange characters, like you get when you look at FileInfo's Utility Thing.
The Thing name is reduced to 20 characters if it is longer - you need to specify the correct Thing name if you want to use or remove it, of course, so you may remove this restriction from the program.

You may have noticed that the Thing linkage contains less information than we list - the type of the Thing is not stored in the Thing linkage but in the Thing itself (th_thing points to the Thing). So we cheated a bit for now and read it from there.

We will have a look at the various structures for the various possible kinds of Things in the next part of the Things series.
■

Q Branch

Shop hours :
Mon - Fri
10am - 6pm
Saturday
10am - 1pm
Wed - closed

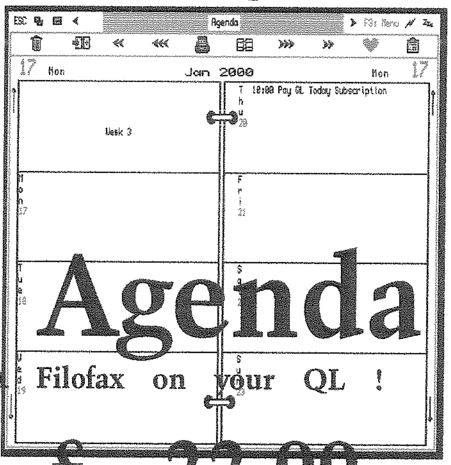
PROGRAMMING

QD 98	£ 53.00
QD + QBasic	£ 69.00
QD + Qliberator + QBasic	£ 110.00
Qliberator	£ 50.00
Master Spy v 3.3	£ 30.00
QPTR	£ 37.00
Easyptr pt 1 & 2 (together)	£ 33.00
Easyptr pt 3 (C library)	£ 16.00
QMake	£ 18.00
QMon / JMon	£ 22.00
Basic Linker	£ 22.00
DISA 3	£ 37.00
QMenu	£ 16.00

UTILITIES

FiFi 2	£ 22.00
QSup	£ 32.00
QSpread v1.44	£ 66.00
Cueshell 2	£ 30.00
Qload / Qref	£ 15.00
Disk Mate 5	£ 16.50
QPAC 1	£ 20.00
QPAC 2	£ 40.00
QTYP 2	£ 30.00
QLQ	£ 32.00

New Program !



Agenda
A Filofax on your QL !
£ 22.00

Text 87
£ 79.00

Typset 94	£ 29.00
Fountext 94	£ 39.00
2488 drivers	£ 29.00
Epson ESC/P2 drivers	£ 26.00

Text 87 is the only QDOS / SMSQ wordprocessor capable of handliing the full screen on the Aurora / QXL / QPC systems. New drivers are currently being written.

'Just Words' by Geoff Wicks
THESAURUS, STYLE CHECK, SOLVITPLUS 3
£ 10.00 ea / ANY 2 PROGRAMS £ 18.00 / ALL 3 PROGRAMS £ 25.00
(Includes Pointer and non-pointer driven versions)
(P.E. versions need Hot_rext, WMAN and PTR_GEN or SMSQ/E to run)
Upgrades from previous versions £ 2.50 + S.A.E. New Manuals £ 1.50

QL2PC - New program !!
Convert text files from QL to PC formats
and much more !
Only £ 10.00
Spelling Crib : PD program £ 1.50 +SAE
or Free if you buy all three programs

We start the new year with some new programs, upgrades and news. Firstly the Q 40 version of the colour drivers is now running nicely with a release looking promising. Fred Toussi has agreed to look into a new version of the drivers to make T 87 work on the Q 40 and there may be other news from that area soon. ProWesS now works on the Q 40 and upgrades to older versions with new instruction are available from us for free, just return the master disk and include postage. We are stocking the new Agenda program from Wolfgang Lenerz and should shortly be offing SuCcess a new Database program. Come along to see us at the Hove or Eindhoven shows for demos of the latest developments.

Youv'e been asking for it and here it is !
The SBASIC / SuperBASIC Reference Manual
The complete definitive guide to BASIC programming in QDOS / SMSQ including three disks of PD toolkits, example procedures and an electronic index.
compiled by Rich Mellor, Franz Hermann and Peter Jaeger

Over 500 pages !
£ 40.00
+ postage

Q Branch

Tel +44 (0) 1903-200212 fax +44 (0) 1903-208070
email qbranch@qbranch.demon.co.uk web : http://www.qbranch.demon.co.uk

Q Branch

Feeling out on a limb ?
Reach out for Q Branch.
Suppliers of Quality QDOS/SMSQ products
Hardware and Software.

The Bank Volt, 6 Coronation Buildings
Ham Road, Worthing.
W. Sussex. BN11 2NN. UK.

ProWesS

ProWesS	£ 48.00
DATAdesign	£ 24.00
Fontutils	£ 30.00
File Search	£ 12.00
PFlist	£ 12.00
Fontpack	£ 60.00
LINEdesign v 2.16	£ 24.00
PWfile	£ 18.00

Paragraph

The ProWesS word processor
Demo version £ 1.50 + postage
Full Registered version £ 18.00
Version 2.02 available now !

SMSQ/E

Gold Card / Atari / QXL Version
£ 76.00
Various Atari versions : call for details
The Colour Drivers are running on the Q Branch Q 40 !
Coming to a computer near you in 2000!

QPC 2 is here !

£ 90.00 (£ 70.00 SMSQ/E Owners)
Upgrades from QPC 1 £ 30.00
(return master disk)
Special offer !
Get Cueshell for only £15.00 with any copy of SMSQ/E

Hardware

We have a small stock of second user items. Auroras / Qubides / Gold Cards / Qplanes / superHermes etc. call us to get details of the items available. These are going fast so call soon.

Recycled superHermes	£ 80.00 *
Recycled Gold Card	£ 60.00 *
Recycled Aurora	£ 75.00 *
Aurora	£ 90.00
Qubide	£ 55.00
Qplane	£ 25.00
Aurora cables	£ 3.00
Aurora rom adaptor	£ 3.00
'Arfa Braquet'	£ 8.00
'Son of Braquet'	£ 18.00
The 'Braquet'	£ 16.00
MC plate	£ 6.50

* when available.

14" and 15" monitors for the Aurora - Call.

Q Branch Programs

The Knight Safe 3 NEW VERSION!	£ 35.00
upgrades from previous versions	£ 5.00
Q - Route v1.08C Route finding programme NEW VERSION!	£ 25.00
Q - Count Pointer driven home accounting	£ 25.00

The Fractal Collection !

This is a brand new program which will produce stunning animated fractal patterns. It will run on anything from a Gold Card to the Q40 and will be capable of using the power of the colour drivers when they are released. Complete with many example files and routines to design your own screens.

Only £ 35.00

Q 40

There are so many extras available for the Q 40 that it is hard to list them here. If you are interested in one of these boards than contact us and we will supply the details. the basic prices are shown here.

Q 40 board with 16Mb RAM & I/O card	£ 330.00
+ SMSQ/E	£ 30.00
Extra 16 Mb RAM	£ 30.00
Tower Case	£ 45.00

All prices do not include shipping.

VISA

We can accept payment by VISA, Mastercard and Switch. You can also pay by Eurocheques made out in Sterling or a Sterling cheque drawn on a UK Bank. Prices include Post and Packing in Europe.

MasterCard

Thierry Godefroy's Masterpiece

Roy Wood

There are those among us who pride themselves on a memory which encomapasses a knowledge of every switch and symbol needed to get the most out of the zip and unzip programs. There is equally a section of the community, of which I am one, who could not even remember how to spell 'unzip'. This is an area in which Fileinfo 2 comes into its own and it is not the only use for this multi-faceted piece of software.

Is it Worth It?

Well, you may ask if FileInfo 2 is worth having but, since it is a free piece of software that question is irrelevant. I personally could not do without it and I was very pleased to see that it performed impeccably on my Q 40.

I first came across this this program when I wrote to Phil Jones (then editor of Quanta) about a completely different piece of PD softawre. Phil sent me a disk with a whole load of programs on it - all of which he considered essential - and suggested I should look into it.

At first I was baffled by the program because I had never used a system which incorporated the concept that you could execute a data file and get the program associated with that file to load up and then load the data. As soon as I had it working I began to see how useful it was and now I would not be able to work so easily without it.

The Big Idea

The whole basis of FileInfo 2 is that you can associate the file extension (those three letter signifiers which are tagged onto the end of files such as '_bas' or '_txt') with the programs which are used to generate, view or otherwise perform certain actions on them. This was taken even further in later versions of the software to the stage that we have now where the current implementation of it surpasses the Windoze version. So how do we use it?

First Base

To start with you do really need to have a program to invoke FileInfo 2. Most of the QDOS/SMSQ file handling programs have hooks built into them to allow FI2 to work. QPAC 2, Cueshell, PWfile and Disk Mate 5 work very well but Thierry Godefroy provides enough programming detail for anyone to write programs to use it so you could do all of this yourself if you feel so inclined.

FI2 needs two things to happen before you can put it to use. Firstly you have to load the program as a resident procedure by either LRESPR'ing it or using the LBYTES command (if you have no Toolkit 2 loaded). Once this is done you must configure the program to suit your system and your needs. This is done using the 'FI2Config_obj' program provided. This is a Qliberated program so you do have to have the QLIB runtimes present and you should also use either the standard QJump config program or Menuconfig to set it up so it knows where the main FI2 program is loaded from and where the online help file is.

Once FI2 is loaded and the FileInfo Config program is configured you can get down to setting it up and the fun starts.

What Can It Do?

The original version had two main functions. The first was the association of file extensions and programs and the second was a '_pic' file viewer. As time has gone by, however, Thierry has continued to tinker with the code and has refined it to it's present level. The first versions of the program would only allow one program to be associated with each extension and this meant that there had to be some kind of compromise when you might want do one of several things with each file.

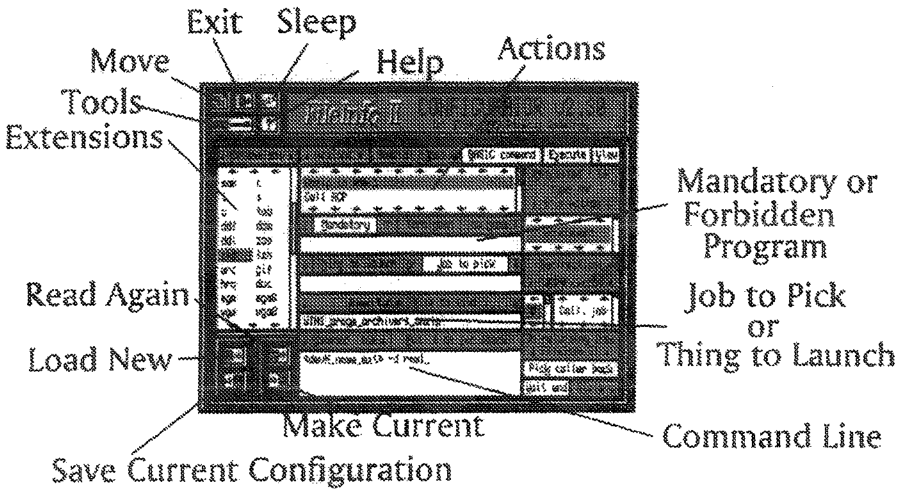
Take a simple SBASIC file for instance. I might want to LRUN it or EX it or load it into QD to be edited or even compile the thing. Last year Thierry obviously had a similar problem so he solved it by providing the user with a menu system which would allow him to choose what to do with the file once FI2 had got hold of it. There is a facility to add the file to an existing ZIP archive via a little piece of BASIC and to cap it all he provided a history device for SMSQ users so you could go back and repeat any procedure that you have already used in the current session.

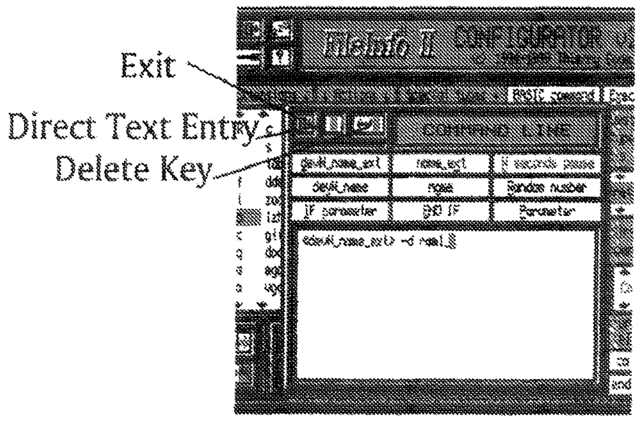
So I Have Loaded It - How do I get the Best Out of It?

The first thing to do once it is loaded is to fire up the FI2 configuration program. If you have configured this correctly it will have online help available by clicking on the '?' icon in the top left hand corner. It should load the current FI2 configuration ready for you to edit it.

The configuration program looks quite complex the first time you see it but don't be daunted by the number of boxes available it is really quite

a simple process so long as you apply a little thought to it. If you have a high resolution display you would be better off loading two copies of the program so you can refer back to the one supplied with FileInfo 2. This one does refer to Thierry's own set up but gives some useful hints on how to set up the more awkward programs and some clues as to which of the setting to use.
At the top left corner of the CONFIGURATOR window you will find the standard Pointer Environment icon for moving the window around the screen followed by a red box with a green arrow projecting from it. This is the 'EXIT' icon. After this is the standard 'SLEEP' symbol. Below these you will see a screwdriver symbol. If you click on this you will get a new menu offering you a host of quick actions. Most of these are quite self explanatory and you can experiment with them all because pressing 'ESC' will cancel the action with no changes made. Beside the screwdriver you will find the '?' symbol which will call up the help file. This can also be achieved by pressing 'F1' and the cursor will change to a handwritten 'Godefroy' signature.
Below this is are the nuts and bolts of the program. The left hand pane is a scrollable, two column, list of extensions with the current one highlit. To select an extension simply HIT on it and the list and the actions set up for that type of file will appear to your right. To add a new extension to the list go to the empty space at the bottom and then HIT the space. A flashing cursor will invite you to enter the new extension.
To the right of the extensions window you will find a series of slots and these are where you enter the actions that have to happen. The top slot is the menu which will appear when you execute a file of the extension mentioned in the left box. If there is only one entry in this slot then that instruction will be performed without displaying a menu but, as soon as there is a second entry, a menu will be called. To edit an entry DO it and a cursor will appear. To add an entry to the list go to the bottom of the list and HIT the space and then type in the new menu item.
Below this is the slot for the 'Mandatory' or 'Forbidden' job name. This simply means that you can set if the actions can only be performed when called from a particular job or should not be performed when called from a particular job. This way you can tailor the menu that appears to conform with the way that you use the system. By way of explanation here I shall use Archivers Control Panel (ACP is another of Thierry's programs which I shall write about in a later issue.)
This program will create archives or extract files from most of the compressed formats such as ZIP, TAR, ZOO etc. The program has FI2 support built into it so, if you DO a filename in a list it will perform the FI2 action associated with it. If you have set the '_zip' extension to call up ACP as one of the options in the menu you would not want it to start another copy of ACP so you can choose the 'forbidden' option for this extension by clicking on the box marked 'forbidden' so it becomes highlit in red. In the slot below you should now have a flashing cursor which will allow you to enter the name of the job from which this option is to made unavailable. This should be the name exactly as it appears in the QPAC 2 'jobs' menu - in this case ACP. If you do a '_zip' file in any program except ACP the


Exit
Sleep
Move
Help
Actions
Tools
Extensions
Mandatory or Forbidden Program
Read Again
Job to Pick or Thing to Launch
Load New
Make Current
Command Line
Save Current Configuration



Command Line Editor

menu should have a 'Call ACP' entry in it. If, however, you do the same thing from ACP this item will be missing.
The next slot is for the 'Thing to launch' or 'Job To Pick'. This means that if you have a resident thing loaded or a job running which fits the name in the slot that job will be used first before a new version of the program is executed. Next comes the name and path of the 'Executable file to load'. This will be the program to launch if the criteria in the box above have not been met.
The last slot we have in this column is the Command line box. This is probably the most complex of all of the boxes to fill in and it comes with its own editor. Here you have to pass a command line to the program to get it to load the data file. This is complex because not every program behaves in the same way and it is not so easy to work out which approach to use. Sometimes you can just note down the keystrokes you use when you load a file into the program manually and use those in this box. For Text 87 you can use the following line:

```
<<pause 5s>>1<<devN_name_ext>>
<<ENTER>><<ESC>>
```

This can be broken down into:

`<<pause 5s>>` start the program and pause for 5 seconds to allow it to load
`1` the keypress to start loading a file `<<devN_name_ext>>` the filename and path
`<<ENTER>>` the same as hitting the ENTER key. Starts the file loading
`<<ESC>>` the same as hitting the ESC key which has the effect of putting the cursor in the editing window

If you want load a file into QSpread you will need:

```
\f <<devN_name_ext>>
```

At the right of the 'Command line' box is another box marked Put it on the job stack'. This will pass the command in the box below to the job stack for processing if it is highlit in red. Text 87 requires that this is not highlit whereas QSpread needs this to be done. You will need some experimentation here before you get all the programs to work correctly but Thierry's example file is a great help.

NEMQLUG

Invites all members of QUANTA and all other QL Users to a

WORKSHOP

On **APRIL 16th 2000**

At **3rd Davyhulme Scout Headquarters**

"The Endeavour", Conway Road, Davyhulme
open from 9.00am, start at 10.00am
AGM to be held at 4.00pm

Programme: From ZX80 to the Q40 - and Beyond

Speakers to include: Tony Firshman, Rich Mellor and Dilwyn Jones

The Scout Headquarters is located close to Junction 9 of M60. Turn towards Urmston along Lostock Road. Conway Road is on the right hand side, go to the bottom and turn right.

There is a Travel Inn at the Trafford centre at Junction 10, which will provide good accommodation. We hope you will bring your partners, who could sample the delights of the Trafford Centre while the rest of us are at the Workshop.

We invite everyone to the Annual Dinner on Saturday April 15th at 7:30pm in Urmston. We anticipate a cost of not more than £16. Please book with Sarah Gilpin on 0161 865 2872 or Fax 01204 371208.

Command Line Editing

There is some help in creating these lines from the little editor window provided. When you DO the box or the command line an editor window will appear which will have some of the more common commands in in a menu above the main window.

There are two modes which can be used in this Window. In the first mode, the one in which you first start, you can select commands from the menu above to build up the command line. There will come a time when you will have to enter letters or numbers directly however and for this you will need to HIT the RED and BLACK button at the top, just to the right of the exit button. This will put you into the direct editing mode and the cursor will start to flash. This mode has a slightly disconcerting feature in that every keystroke that you make appears in the window. If you make a mistake and hit the delete key or the cursor keys you will get CTRL and an arrow for 'delete' or just an arrow for the cursor keys. If you want to leave the direct editing mode or actually delete part of the entry you must first have to move the cursor out of the window for a couple of seconds until the cursor stops flashing. To delete an item HIT the icon on the top left with the red square and the green arrow beside it. Once you have finished editing the line you can go back to the main program by using the usual exit symbol.

Other Options

On the far right side of the main windows there is a slot in which you can set the data type which FI2 should expect for the action selected. This can prevent unexpected actions being performed on a program which has a misleading extension.

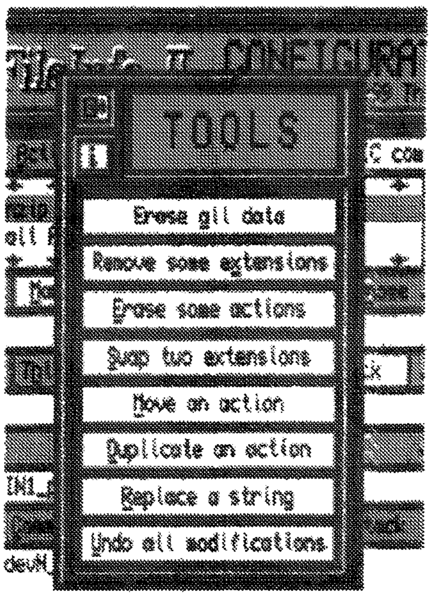
Below this you can set the number of re-directions it will expect and, finally what FI2 should do when the task has been completed, and if the action should be entered into the history. For most tasks these can be safely ignored and the history set 'on'.

Save It!

Once you have finished configuring the program you should save it but there are a couple of tips here I should pass on. The first one is pretty simple - but vital. If you already have a working version of FI2 make a backup of the FILEINFO2_BIN file before starting work on a new version this way you can always go back if you mess up. You should also keep a copy of Thierry's version to use for inspiration.

At the bottom left corner of the main window there are four boxes.

The first of these will load a copy of the file from disk which will allow you to run two copies of the configurator- one with your settings and the other with either your old settings or someone else's. It will also allow you update to a new version of FI2 without losing the old settings. The next box will read in the settings of the currently loaded program again useful if you have been editing and lost track of what you are doing. The bottom left box will save the current settings to disk creating a new Fileinfo2_bin file and overwriting the old one. The last box will make the settings you have just created current - ie the program will behave in the way you have just set it to but the file will not be saved or overwritten. This last box is very useful because it is a 'try before you buy box'. You can set up a new menu or configuration and then make these current so you can try it to see if it works.



Tools Menu

Updating from a Previous Version

Updating from a previous version is easy. All you have to do is to LRESPR the latest version of the program into your machine and then use the load button to read in your previous settings. If you then save the file you have successfully updated your copy of the program.

History Lesson

There is a FI2 history available as a menu to all those who have the menu extensions loaded and are using SMSQ/E. This can be called up by setting up a hotkey using a line in your BOOT file:

```
1080 ERT HOT_THING ('f',
'FileInfo II thread';'-h')
```

next time you press ALT/f you will get a menu showing all of the actions you have performed in the current session and by HITting one of the items perform that action all over again.

QASCADE and FI2

Qascade has FI2 support as well. This means that you can set up an action for, say, QSpread speadsheets. Then use the line:

```
FI2 Foreign Prices Win1_DATA_
Foreign_prices_tab
```

in the Qascade_RC file. This will give you an entry in the Qascade menu which will read as 'Foreign Prices' and when you click on it FI2 will load QSpread and then load the selected file. I do this for all of my commonly used files and it saves a lot of typing. Another thing that saves typing is setting up a line in FI2 for '_zip' files. Try this:

```
<<devN_name_ext>> -d ram1_
```

and set the 'Executable file to load' option to point at the location of the UNZIP program. This will unzip the file directly to ram1_

Summary

I have only really scratched the surface of this clever and useful program. There are many more things that can be done especially by writing short BASIC programs to be executed from FileInfo 2. Maybe there is a competition there?

Well done Thierry!

The QL Family Tree - The 80's

Dilwyn Jones

A look at the history of the QL from its launch in 1984 to the present day.

1984 April sees the delivery of the first batch of 'dongled' QLs, machines with their operating systems in EPROMs hanging from the back EPROM socket. All sorts of problems with the machine and its software lead to scathing reviews in the press. Recognising the need for a user group, Leon Heller and Brian Pain had already (back in February in fact) formed IQLUG, the Independent QL Users Group, which was later to change its name to Quanta because many people could not handle the pronunciation of IQLUG! Sinclair rapidly releases several updates to the QL ROMs, with versions JM and JS becoming the commonest 'standard' versions (the JS appeared in early 1985). Psion also releases eagerly awaited updates to the four programs supplied with the QL. GST, the company originally contracted to produce the QL operating system releases its OS without a BASIC interpreter under the name of 68k/OS as a plug in board in Autumn 1984. Sinclair had used Tony Tebby's QDOS operating system for the QL, which included a SuperBASIC interpreter written by Jan Jones. Miracle Systems Ltd was a small startup company which began supplying serial-parallel converter printer leads to fill a gap left by Sinclair's decision not to include a parallel port on the QL. You could buy QLs and peripherals in high street stores (such as W.H.Smith in Britain). The first QL magazines begin to appear, with some dispute over who has rights to the early magazine names such as QL User, which happened to clash. Sinclair had launched QLUB, a kind of club for QL users, which issued a regular black & white newsletter with news of new products etc. A development of QL technology (including microdrives) was to be used in the ICL OPD (One Per Desk) computer. Tony Tebby resigns from Sinclair Research and sets up QJump to develop his own QL products.

1985 QL World magazine is launched late 1985 and lasts 5 or 6 issues before merging with QL User in early 1986, but retains the 'QL World' name. February sees the price of microdrive cartridges cut from a whopping £4.95 to £1.99. The price of the QL itself is cut to £199 from 2nd September 1985, and Sinclair announce plans to badge printers and a disk system from Micro Peripherals. The first icon controlled environment software (ICE) is published by Eidersoft, while in the autumn CST were to release the first QL hard disk system in a range of capacities from 10MB to a (then) massive 40MB! Toolkit 2, a successor to Toolkit 1 sold by Sinclair, is available from Tony Tebby (he also produced the original Sinclair Toolkit). Psion release classic programs such as the Psion Chess which included a novel 3D board display option. Robert Maxwell apparently bought a 12 million pound stake in Sinclair. Modems became fairly commonly available for the QL, software development gave a rush of new software and one of the most famous of QL software companies, Digital Precision Ltd, started advertising and released the first QL basic compiler, Supercharge, which annoys a lot of people through its use of the dreaded Lenslok protection system - I know, I always had difficulty with mine! Rumours abounded that Sinclair might release a plug in ROM card containing Xchange for the QL (which didn't happen in the end). Companies such as Simplex, Quest, CST, Micro Peripherals, Silicon Express, PCML, Medic, Technology Research Ltd, Sandy and Miracle Systems were releasing ever more hardware for the QL. Miracle Systems Ltd later were to become perhaps the best known third party peripheral producer and the longest surviving of those early companies. By the end of the year, they had produced the Expandaram memory expansion unit with a choice of memory from 256K to 512K. TF

Services, another early QL company which has lasted to this day, started selling its computer power cleaners which were claimed to reduce the instances of crashing QLs by improving the mains power to the computer by removing spikes and surges.

1986 QL World and QL User merge. The spring sees the sale of the QL and Spectrum to Amstrad, who immediately discontinue production of the QL. Two organisations announce plans to produce QL successors. CST produce the Thor, based on original QL circuit boards, while Tony Tebby announces plans for a 'QLT' which was later to be called the Futura and to be produced by Sandy UK PCP Ltd, though it never went into production apparently because of a decision by the Italian parent company. Sandy eventually release the QXT640, a reboxed QL with memory expansion and disk interface. Autumn sees the launch of the QLiberator compiler by Ian Stewart and Adrian Soundry of Liberation Software and some heated discussion follows on whether QLiberator or SuperCharge is best! In May 1986 QL World reports that a new computer called the MicroBox III would use a derivative of QDOS as its operating system - called SMS. This OS would later be adapted and released as a cartridge for the Atari and in time would develop into the SMSQ we know today. GAP Software releases Front Page, the first desktop publishing program for the QL. Miracle Systems released a tiny plug in modem-on-a-lead for the QL which incidentally was not B.T. approved at the time! Sector Software, who were to become a major QL software publisher, release Taskmaster at the November 1986 ZX Microfair. Taskmaster was one of a number of task-switching aids for the QL, claiming to allow the user to make easier use of multitasking (for multitasking read task-switching). Companies such as Schoen produce add on keyboards to replace the standard QL keyboard.

1987 The 'QL User' name disappears from QL World's cover with the February issue. A Strip Poker game featuring "Denise" published by Talent raises a few eyebrows on the QL scene. Digital Precision announced the Turbo compiler, while author Simon Goodwin went on to release Speedscreen, the first screen output acceleration software for the QL. Sector Software announced a number of innovative programs, including Spellbound (a real time Spellchecker which actually checked spelling as you typed) and Flashback, a fast database system. CST announced they were big in Denmark, and released first the Thor 20, a 68020 based derivative of the original QL-based Thor, and later the Thor XVI, a new 68000 QL compatible computer. CST's O.S. was called Argos, a derivative of QDOS adapted for the Thor. A German company made available a version of the CP/M operating system for the QL, while Dutch company Data-Skip advertised a wristwatch databank by Seiko which linked to the QL via a cable. QJump release the QIMI mouse interface, while Sandy build a compatible system into their SuperQBoard expander. Miracle Systems took the standard QL expansion of disk interface and 512KB extra RAM a stage further with the classic Trump Card (the term had in fact been used by another QL company in its advertising some time before!) which pushed memory up to a massive 896KB by using all available address space, and included a disk interface, screen dump software and Toolkit 2 on board. Although a dead end in expansion terms, having used all available address space, the Trump Card proved extremely popular and quickly became a best seller. Text 87 announced to become the first WYSIWYG word processor for the QL. Simon Goodwin sets out to document known QL ROM bugs in articles in QL World, a task he was to continue for some time through various articles over the years.

1988 The Swedish QL Users Group alone claimed a membership of over 400 in 1988 - compare that with Quanta's current membership! By now there were strong QL user groups in many countries, from the USA to Europe. A number of major QL enhancements were announced in this year, some of which came to market, some of which didn't. One which did was the first Atari ST QL emulator boards. Originally produced by Futura Datasenter in Norway, it was briefly produced by Strong Computers before being acquired by Jochen Merz, whose adverts were to appear in QL World at the end of the year - including an advert for the now famous QD editor. Jochen should have known better than to get involved with the ST-QL emulator, since one of the first he got wouldn't fit his computer and he ended up having to drive 2,600 km (yes, 2,600) to Norway via Denmark to sort it all out. If you can afford an Audi Turbo, make use of it I suppose! Jochen had in fact lost a lot of money to other companies in the time before this, companies which included Ultrasoft in Germany where one of the partners (David B. Smith) decided to leave, and formed a separate QL company (Hallmark Software), while Martin Bernd continued with Ultrasoft. That is the story of how Jochen Merz Software came to be. And Jochen would maintain his luck is not much better to this day. Gigasoft's Megaram expansion unit replaced the QL's 68008 processor and allowed up to 3MB of RAM. Sandy announced an ultimate QL expander called the Megaboard, which never really made it to market. Neither did a prototype 68010 add-on board from Power Computing for PCs with promised SMS-2 operating system - this would have been the first QL on a PC card emulator, well before the QXL. Miracle produced the QL Midi pack - a Midi interface for the QL. TF Services sold a barcode reader for the QL while a new second processor chip appeared which claimed to fix the keyboard bounce problems afflicting some QLs. While succeeded in that respect, the device could also cause serial communication problems for those using modems. QL Adventurers Forum magazine was launched by CGH Services, reflecting the interest in games on the QL at the time. Bill Johns launched Club QL International as a self help group for less experienced QL users. QPAC1 was announced and Digital Precision launched the Lightning screen accelerator, and quietly dropped royalty requirements for programs sold compiled using Supercharge, while launching Supercharge Special Edition. Concerns arise about the future of CST and the Thor computers as they move operations to Denmark, while things look like turning nasty in Spring 1988 when an apparent tiff between Tony Tebby and Digital Precision results in DP publishing an advert in QL World May 1988 where DP apologise with the sentence "We apologise unreservedly to Tony Tebby and regret any upset to him and his family and any damage to his reputation" following comments made in connection with compatibility between DP's compilers and Tony Tebby's products. Other controversy arose in the spring at the Quanta AGM, which led to co-founder Brian Pain's resignation following a heated debate about finance. Co-founder Leon Heller was replaced as editor of the newsletter by Roy Barber, though Leon stayed on as a committee member for some time. In Berlin, Germany, Rainer Kowallik was working on the first version of the Amiga QDOS emulator. Belgian software house PROGS released The Painter (after two years of coding!) a pointer driven art package written in machine code. They later went on to release Line Design, Prowess and Proforma that we know today

1989 This year sees the launch of a different type of emulator - the QL emulating a DOS PC. Digital Precision first launched The Solution and later in the year launched an improved version called PC Conqueror. At the same time, two other such emulators were published by Ant Software and Schoen, though DP's offerings were the only ones to achieve great success. PDQL launched a program called Basic C-Port to allow users to translate their SuperBASIC programs to C which was becoming increasingly popular among QLers, especially as it made it easier to port QL programs to C compilers on other computers. Hi-Soft released Hi-Soft Basic, a compiler for the Atari ST and Amiga developed from Supercharge under licence from Simon Goodwin. Miracle Systems launched their QL hard disk system for £399, which plugged into the EPROM slot behind the QL. Rebel Electronics also had a hard disk system for the QL, and the operating software for this system was later adapted for the Qubide interface. Another hard disk system was available from Germany, but the poor microdrive cartridges were in trouble as Ablex (who manufactured them) had to face up to a shortage of suitable tape and declining sales making it less viable to continue producing them. Focus announced plans to make QL World a subscription only magazine, but later postponed the plan. QView releases the first Minerva ROMs by the team of Laurence Reeves, Jonathan Oakley and Stuart MacKnight. Complaints begin to appear about Super User Bureau, an organisation set up to support the QL but whose newsletters seemed reluctant to appear as often as they should. QL Technical Review magazine from CGH Services makes an appearance. The first working versions of the Amiga emulator were in use, version 2.00 June 1989.

Next issue, we have a look at the 90s.

BYTS OF WOOD

SAW POINTS OFFCUTS AND SNIPPETS

Well here we all are in the 21st century still living and relatively unbugged. I say relatively because although we were all sitting around smugly saying 'our system won't suffer from this silly Millennium thing' operators of the PBOX database system found themselves caught out by millennial madness and having to deal with corrupted dates.

Accusing fingers were pointed at the 'C68' code (well we did inherit it from another system) but it seems, from reading the 'maus.computer.ql.intl' mailing list that the C68 code could have handled it all quite well had anyone actually noticed it was wrong. Complacency breeds - well - more complacency really. Since Phil Borman, author of the PBOX software, is no longer an active QL programmer Jonathan Hudson has offered to ride to the rescue and fix the problem but Phil already acknowledged his errors and posted fixes on his web page.

Quanta also fell into the Y2K trap because the Quanta Accounts are kept on an old program which will not start if the date is wrong and it thinks that it is so, at the time of writing

this, they are locked out of their own system. This is a program written by Sage many years ago called QL Integrated Accounts so, if you are using this as well, beware!
Apart from that we QLers have escaped unscathed. Our Big Bad Bug comes in 2097 (I believe) and Mark Knight has said that if anyone has a problem with his software in 2097 and can get in touch with him he will fix it for free. Can't say fairer than that can you ?

Compute me a Rainbow

I usually try to avoid mentioning the programs or hardware that we sell at Q Branch in these articles because I do not want it to seem like advertising but I have spent many hours on the Q40 in the last few weeks and I am beginning to do more and more work on it. One of the many fascinating things about is the ability to program in so many colours now the first beta versions of the colour drivers have been released. OK I know that the PC has had these colours for ages but the PC is such an impersonal beast and, more to the point, is so inaccessible to simple programming. If you want to write something simple on the PC you need various packages like the Visual Basic Studio etc. whereas it is all there in SMSQ and you can even do it from the command line.
As soon as the drivers were installed I began to try out various bits and pieces to see what would work and what would not. Some of the early problems have now been solved and there are many programs which ran straight off with no problems at all. There are oddities however. Text 87's way of using the display has already caused problems on the Aurora which needed a patch from Phil Borman but, once again, it seemed to be unhappy in the new colour driver environment. Tony Tebby was looking into setting up a special mode for programs like this to run in but, after an email to Fred Toussi, the author of Text 87, I got a lot of support. He is now looking into ways of getting Text 87 to work under the new drivers. It may also be that we can persuade him to do a re-write of some of the routines and come up with a new version. This would be most welcome since most of my letter writing and all of our manuals are done in T87. Jochen is in the same position so we will work at it.

Basic Colours

One thing trapped me more than I expected and that was playing around with some fairly basic BASIC. I wanted to put a shadow under a text block in a program I had been writing. In the old 4 colour SMSQ this is a simple block command but, with several shades of grey available, I got involved in trying to find the most realistic way of displaying a shadow. Simple really but so engrossing to watch the effect of different sizes and colours of borders that I got lost in the fun of it all and hours had gone by by the time I had finished. Of the final result was the one I had decided was the most logical to start with but....
And, of course, the next thing to do was to write a display to show off the colours for the next workshop.

Basic Sounds

After this explosion of colour the QL-Users newsgroup came up with a few comments about the use of sound files on the Q 40. For those of you who don't know the Q40 has a fully functioning stereo sound system and this means you can play back sounds on it if you have connected some kind of speaker system. The only sound format so far supported was '_ub' (unsigned bytes) and this is not found very often in the normal PC environment. Most files on a PC are either MIDI (musical Instrument Digital Interface or Mindless Idiots Desire It) or '_wav' files. I have previously mentioned a program called 'AWAVE' which can be used to convert from other formats to '_ub' but this has to be run on a PC and most people would prefer to avoid that.
The comments on the board led Jonathan Hudson to explore a couple of sound programs in the public domain and he ported one over and this will run on the Q40 to produce the same effect.

Blow your SOX off

The program is called SOX and does not stop at mere conversion. You can add all sorts of digital effects as well. Hours of annoying the neighbours there I think. Having done this Jonathan began work on a sound player for the Q 40 and we have been testing this out too. This may, eventually lead to a program that could be able to play all sorts of sound files on the Q40 so QL workshops just got noisier.

The ProWesS Factor

The Q40 has inspired Joachim van der Auwera to do some work on ProWesS and he now has a working Q40 version. This has been a major factor in my adopting the Q40 as an integral part of my system instead of a toy. Since I prepare all of my adverts on LINEdesign as soon as a high resolution display became available for the Q40 I began to export the graphics files to it in order to speed up the process of making the ads. The speed and memory power

of the Q40 mean that things are drawn quicker and there are less passes needed to print it out.
One interesting thing which I found when I started to set ProWesS up to run on the Q40 was that it was already able to use the extended colour scheme. It therefore becomes the first QL program to actively use them. Again more time spent in front of the screen while I configured different colours to see what it would look like.
ProWesS really comes into its own on this system since everything moves along with a speed and fluidity which it a pleasure to use.

All LINUX'd Up

Reports from Peter Graf say that there is a LINUX installation Cd rom available for the Q40. I will be looking into this and more details will be available soon but the good part about this is that it will boot directly from the SMSQ/E part of the Q 40 and give the user access to all of the LINUX programs including Netscape and the TCP/IP stacks. This will mean that Internet access is now available on a QL compatible machine without having to go over to a PC.
For many people the only thing they needed from a PC was the ability to access the Internet and this could be an ideal solution to their problems. the Q40 will run everything QDOS/SMSQ that they need and they can do the communication part via LINUX. There are many things available for LINUX now you could probably do a lot with this installation. The CD will apparently be available with a floppy disk to get the basic CD rom drivers onto the LINUX partition of the hard disk so that the CD can function to get the whole thing up and running.

Catch a Fire

Tony Firshman and I took a little jaunt down to Gatwick Airport shortly before Christmas to meet up with Zeljko Nastasic who was having an overnight stop between Croatia and the U.S. Apart from the social factor of meeting up we wanted to discuss the development of the GoldFire now that Qubbesoft are no longer involved.
All three of us were very much in agreement that it should continue to be developed although the progress on the work has been considerably hampered by lack of time on Nasta's part and his move to the U.S. There has be some conceptual development, however, because some new chips have emerged which has meant that the number of chips on the main board will be considerably reduced. The new design is still to be finalised and I doubt if the prototype will emerge quickly because Nasta still has to fix work in America and go through the business of settling down in a new country but he did say that once all of his stuff had arrived the initial design work would only be a matter of weeks. This is, of course weeks of work not linear time, but that is encouraging. I am even more encouraged by the fact that he still has a commitment to the development of the system.
Even when the main card design has been sorted out we will still have to tackle the other plank of the project and get the drivers written. The board will be completely different from the Super Gold Card which at least had the Gold Card as a starting point. In the case of the GoldFire Nasta seems sure that the floppy and parallel port drivers could be the same as those on the Q40 (with a bit of a tweak). There are some advanced features on the chips but those could be integrated at a later stage. The board should have an in built keyboard and mouse interface. Nasta does not say how this would work with an Aurora which would need something in the place now occupied by superHermes - maybe it would work alongside it.
The code that would need writing would be the startup and ram test. The GoldFire is projected to have in excess of 16 Mb (Nasta did talk of an upper limit of 128Mb) so a whole new routine would be needed.

A Huge Plus

Mentioning all the above hardware things has lead me into thinking about the major differences in concept and execution between the QDOS/SMSQ system and the PC way of doing things. With a QL there is very little you can do to stop it working completely unless you remove chips or something similar. OK you can trash the odd hard disk by writing into its map area but the system will still fire up. Even writing a boot file such as

```
100 Print 'My Boot'
110 GOTO 100
```

on the hard disk can be circumvented by putting another disk into flp1_ and booting from that.
The PC on the other hand can break apart if you breathe on it. People come into our shop with hard disks which will not fire up because the system files have disappeared or maybe someone has fiddled with the BIOS and turned the hard disk off or any manner of other problems many of which can only be resolved by a format and a complete re-installation of the O/S. One thing that always makes me laugh is the way that often a screensaver, which is supplied as part of Windoze

TF Services

superHermes

A major hardware upgrade for the QL

All Hermes features (working ser1/2, independent baud rates/de-bounced keyboard/keyclick) PLUS full 19200 throughput on ser1/ser2 not affected by sound IBM AT keyboard interface (plus foreign drivers) // HIGH SPEED RS232 industry standard two-way serial port. 4800cps throughput (supergoldcard - qtpi - zmodem) at 57600bps // THREE low speed RS232 inputs (1200 to 30bps)- incl SERIAL MOUSE driver. Other uses include RTTY/graphics tablet etc // 3 spare I/O lines (logic) with GND/+5V // Capslock/ scrollock LED/Turbo/keylock connectors // 1.5k user data permanently in EEPROM

All this on a professional board about twice the size of the 8049 co-processor it replaces

Cost (including manual/software)	**£90** (£92/£87/£90)
IBM AT UK layout Keyboard	**£22** (£24/£23/£27)
Serial mouse	**£11** (£13/£12/£14)
Capslock/scrollock LED	**£1** (£1.50/£1/£1.50)
Keyboard or mouse lead	**£3** (£3.50/£3/£3.50)
High speed serial (ser3) lead	**£4** (£4.50/£4/£4.50)

superHermes LITE

All Hermes features (see above) + an IBM AT keyboard interface only. Entry level superHermes.

Cost (incl keyboard lead)...**£53** (£55.50/£51/£53.50)

Minerva

MINERVA RTC (MKII) + battery for 256 bytes ram. CRASHPROOF clock & I^2C bus for interfacing. Can autoboot from battery backed ram. Quick start-up.

The ORIGINAL system operating system upgrade

OTHER FEATURES COMMON TO ALL VERSIONS

DEBUGGED operating system/ autoboot on reset of power failure/ Multiple Basic/ faster scheduler- graphics (within 10% of lightning) - string handling/ WHEN ERROR/ 2nd screen/ TRACE/ non-English keyboard drivers/ "warm" fast reset. V1.97 with split OUTPUT baud rates (+ Hermes) & built in Multibasic.

First upgrade free. Otherwise send £3 (+£5 for manual if requd). Send disk plus SAE or two IRCs

MKI...£40 (£41/£40/£43) **MKII...£65** (£66/£63/£67)

QL REPAIRS (UK only)

Fixed price for unmodified QLs, excl microdrives. QLs tested with Thorn-EMI rig and ROM software.

£27 incl 6 month guarantee

QL RomDisq

Up to 8 mbyte of flash memory for the QL

A small plug in circuit for the QL's ROM port (or Aurora) giving 2, 4 or 8 mbytes of permanent storage - it can be thought of as a portable hard disk on a card, and reads at some 2 mbytes per second. Think of it - you could fully boot an expanded QL, including all drivers/SMSQ etc off **RomDisq** at hard disk speed with only a memory expansion needed.

2 mbytes RomDisq	**£39** (£41£37/£40)
4mbytes RomDisq	**£65**(£66/£63/£67)
8 mbytes RomDisq	**£98** (£100/£95/£99)
Aurora adaptor	**£3** (£3.50/£3/£4)

MPLANE

A low profile powered backplane with ROM port

A three expansion backplane with ROM port included for RomDisq etc. Aurora can be fitted in notebook case and powered off single 5V rail - contact QBranch for details. Two boards (eg Aurora and Gold Card/Super Gold Card/Goldfire fixed to base. Suitable for Aurora (ROM accessible from outside) & QL motherboard in tower case. Specify ROM facing IN towards boards, or OUT towards back of case.

Cost...**£34** (£36/£33/£35)

I2C INTERFACES

Connects to Minerva MKII and any Philips I^2C bus

Power Driver Interface **16 I/O lines with 12 of these used to control 8 current carrying outputs (source and sink capable)**

2 amp (for 8 relays, small motors)	**£40** (£43/£38/£44)
4 amp total (for motors etc)	**£45** (£48/£43/£50)

Relays (8 3a 12v 2-way mains relays (needs 2a power driver)...**£25** (£28/£23/£27)

Parallel Interface Gives 16 input/output lines. Can be used wherever logic signals are required..**£25** (£28/£23/£27)

Analogue Interface Gives eight 8 bit analogue to digital inputs (ADC) and two 8 bit digital to analogue outputs (DAC). Used for temp measurements, sound sampling (to 5 KHz), x/y plotting.....**£30** (£31.50/£29/£30)

Temp probe (-40°C to +125°C)	**£10** (£10.50/£10/£11)
Connector for four temp probes	**£10** (£10.50/£10/£11)
Data sheets	**£2** (£2.50/£2/£3)

QL SPARES

Keyboard membrane	**£24** (£25/£24/£27)
1377 PAL	**£3** (£3.50/£3/£4)
Circuit diagrams	**£3** (£3.50/£3/£4)
68008 cpu or 8049 IPC	**£8** (£8.50/£7.50/£9)
8301/8302 or JM ROM or serial lead	**£10** (£11.50/£10/£11)
Power supply (sea mail overseas)	**£12** (£17/£16/£21)

Prices include postage and packing (Airmail where applicable). Prices are: UK (EC/Europe outside EC/Rest of world). Payment by cheque drawn on bank with UK address, debit card/Mastercard/Access/Eurocard/postal order or CASH! (No Eurocheques). SAE or IRC for full list and details

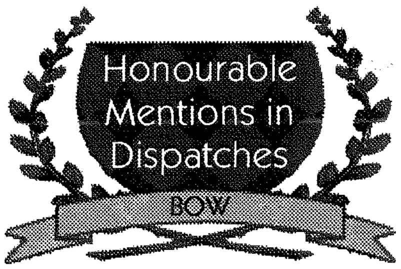
26 Sep 99

VISA

29 Longfield Road, TRING, Herts, HP23 4DG
Tel: 01442-828254 Fax/BBS: 01442-828255
tony@firshman.demon.co.uk http://www.firshman.demon.co.uk

MasterCard

and written by the same company can crash with a message 'This program has performed an illegal operation and will be closed down'. What the hell does that mean? What am I supposed to do about it? Has the program robbed a bank, written slanderous things about Bill Gates or been caught behind the bike sheds having a quick puff on some wacky baccy?
True the PC system is a far more complex beast than ours but I am happier doing things on my SMSQ system any day.



The Honourable Mention this issue is won by Marcel Kilgus for his stirling work on QPC II v1.53.
The major problem people have had when running QPC II is to transfer data between the Windoze section and QPC II. Since the majority of this data is text based the new version of QPC II solves the problem admirably by installing a link between the Windows Clipboard and the Scrap function in the Menu Extensions.
All you have to do is to make sure you have the Menu Extensions loaded and add the command

```
QPC_SYNCSCRAP
```

either to the boot file or via the command line. I leave it in the boot file because then I don't have to think about it.
Once you have done this all you have to do is to go to any program in QPC that supports scrap and load a block of text into it. Once you have done this use the ALT-TAB key combination to go back to Windoze, open a word processing package and use CONTROL-V to insert the text. To use the process in reverse send the text to the Windoze clipboard by marking it and using CONTROL-C and then go into QPC II. You will find the text you have marked in the scrap.
Not all programs support the scrap facility but you can use Phil Jones' excellent 'Scratch' program to make them do it. 'Scratch' is a PD program which was released by Phil a while ago. I load it as part of my normal boot file and it puts a little double height icon in the button frame. You will recognise the icon because it is the familiar 'clipboard' symbol found in QD and QSpread.
When there is no text in the scrap the clipboard is blank but, as soon as it has something in it, the symbol will change from a blank board to one which has symbols on it. If you click on this it will change to an animated cursor which you can position over an open text window, Quill, Perfection or Text 87 for instance and the text will appear there.
Of course some text programs don't allow you to export to scrap but Duncan Neithercut's ClipScrapBoard program will fill that hole so you have it all.
Phil's work deserves a belated Honourable Mention because this little program is so useful. It will not only perform the above actions but will also enable you to easily clip a section of the screen as a pic file and even recolour it.
The Menu Extensions have the ability to manipulate the 'scrap' from SBASIC so you can write your own programs to access this. See the documentation supplied with the QMenu package.
One thing you should know about this is that it is important that you load the Menu Extensions before you invoke the QPC_SYNCSCRAP command or an error will occur.
Both Phil's and Duncan's programs can be obtained from the usual PD sources.

A Quick Word about Spreadsheets

Oh and by the way I learned something about speadsheets which is worth a mention. After talking, in an earlier Byts of Wood, about transferring data to the Micro$oft database 'Access' I learned it was even easier to transfer QSpread spreadsheets into 'Xcell'. All you need to do is to print the sheet to a file and then save the file with .txt extension on a DOS disk. All you have to do is to import the 'text' file into Xcell and it is immediately organised as a spreadsheet. You do lose the formulae but it is a good start.

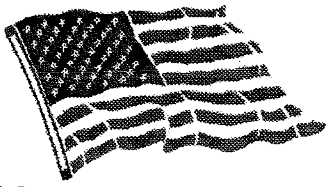
Peering Into The Haze.

At this time of year there is a great temptation for magazines to look forward into the future and, being on doorstep of a new century that is even more tempting. The QL and its emulators have had a great past and thing are showing signs of perking up a bit these days. Jochen has just released Wolfgang Lenerz's 'Agenda' diary program and a new database program (mentioned in a previous column) should be on its way very soon. The Q 40 is gaining converts to its high speed computing and the colour drivers are finally in the beta test stage. We may have fewer users now than we did before but there are also people coming back to QDOS/SMSQ via the excellent emulators now available. We may not be able to play DOOM III but it is still more fun to write a bit of BASIC on a QL than fiddle with DLLs and virtual device drivers on a PC.

A Happy New Century to you all.

US QL 2000 Show

Sponsored by NESQLUG



White River Junction, Vermont USA on May 20, 2000

The show will be held at the Hotel Coolidge in White River Junction, Vermont USA on May 20, 2000. This small town is right on the border of the states of New Hampshire and Vermont in the heart of the US region known as New England. Close to major Interstate Highways I-89 and I-91.

The Hotel Coolidge has a web site **http://www.hotelcoolidge.com/** and can be contacted by email (**hotel.coolidge@valley.net**). There is also an 800 number that will work within the US and Canada (800-622-1124) and a regular number (802-295-3118). If you call please mention NESQLUG or QL Computer Show to get special rates:
Single $49   Double $59
I have reserved 8 rooms to be held until April 20th but will reserve more if we need them. All rooms are non smoking. The hotel has a restaurant and bar. For those interested in the experience or on a tight budget will welcome travelers to stay in our home. Just let me know.

As in past years on Friday evening, May 19, at 6 PM we will meet in the hotel lobby and go to dinner together. The show will be at the hotel from 10 AM until 4 PM on Saturday, May 20. A lunch will be served during the show. Admission to the show including lunch will be $12 per person but traders will be free. After the show there will be a dinner at the hotel which will be $20 per person. On the following day, May 21, there will be an all day gathering at our home, Bill Cable/Mary Boyle, for the day. It is located in Cornish, New Hampshire which is about 15 miles from the Hotel Coolidge. See the Wood And Wind Wind Generator in action.

Please contact: Bill Cable Director of NESQLUG at **cable@cyberportal.net**or (603) 675-2218 if you have any questions. The Hotel Coolidge Web Site has a map and more detailed maps and directions will soon be available at the NESQLUG Web Site. If you are planning to come please let me know so we can keep an accurate list of who will be there. Hope to see you there.

Airport	Driving Time to Show Location	
Boston, MA Airport	3 hour drive north	
Manchester, NH Airport	1 1/2 hour drive north	Nice Drive
Hartford, CT Airport	3 hour drive northeast	
Burlington, VT Airport	2 hour drive east	Nice Drive
Montreal, Canada Airport	4 hour drive south	Nice Drive
New York, NY Airports	6 hour drive northeast	
Lebanon, NH Airport	10 minutes drive	

this is small regional airport has rental cars

The QL Show Agenda

19. Feb. Eindhoven, The Netherlands
St. Joris College, same venue as always.

27. Feb. Hove, United Kingdom
Excelsior Hotel, details below.

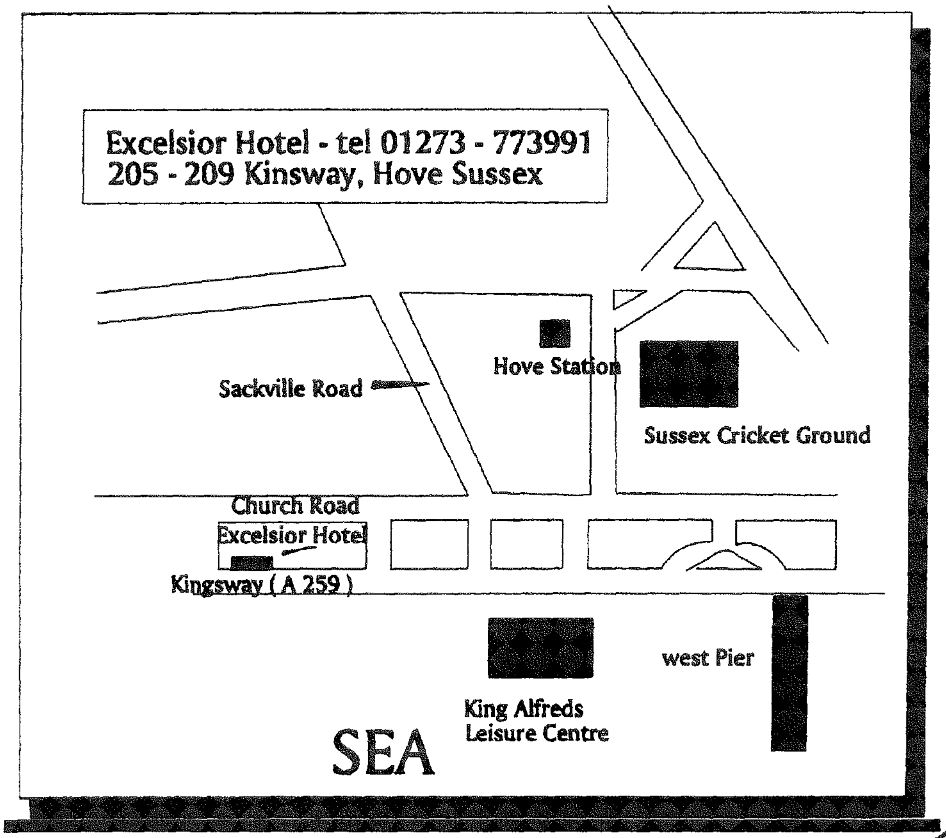
16. April - Davyhulme, United Kingdom
NEMQLUG Workshop & Quanta AGM
Details on page 51!

13. May Eindhoven, The Netherlands
St. Joris College, same venue as always.

20. May East Coast US Show. See reverse side!

Sussex User Group / Quanta Workshop
Sunday 27th February 2000 - 10am till 4pm

The Fourth Sussex User Group / Quanta Workshop will be held in the Excelsior Hotel on **Sunday 27th February 2000**
There will be talks, demonstrations, a 'Bring & Buy' stall, traders and users.
Come along and see all types of QDOS/SMSQ software and hardware in operation. Cheaper rates are offered for the weekend at the hotel if you quote 'Quanta' when booking a room.
All this and a day by the sea too - how can you resist?


Excelsior Hotel - tel 01273 - 773991
205 - 209 Kinsway, Hove Sussex
Hove Station
Sackville Road
Sussex Cricket Ground
Church Road
Excelsior Hotel
Kingsway (A 259)
west Pier
King Alfreds Leisure Centre
SEA